Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

Пояснювальна записка

до магістерської кваліфікаційної роботи

на тему «Адаптивний метод автентифікації користувачів мобільних пристроїв»

08-20.МКР.002.00.000 ПЗ

Виконав: студент 2 курсу, групи 1БС-18м
Спеціальність 125 Кібербезпека
ОПП Безпека інформаційних і комунікаційних систем

_____ Вітович М. М.

Керівник: к.т.н. ст. викл. каф. ЗІ

_____ Лукічов В.В.

Рецензент: к.т.н., доц., доц. кафедри ОТ

_____ Крупельницький Л. В.

Вінниця - 2019 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації
Освітньо-кваліфікаційний рівень магістр
Спеціальність 125 Кібербезпека
ОПП Безпека інформаційних і комунікаційних систем

ЗАТВЕРДЖУЮ
Завідувач кафедри ЗІ, д. т. н., проф.
_____ В. А. Лужецький
_____ _____ 2019 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Вітовичу Максиму Миколайовичу

1. Тема роботи: «Адаптивний метод автентифікації користувачів мобільних пристроїв»
керівник роботи: Лукічов Віталій Володимирович, к.т.н. ст. викл. каф. ЗІ, затверджена наказом ректора ВНТУ№ 254 від 02.10.2019 р.
2. Строк подання студентом роботи _____ 2019 р.
3. Вихідні дані до роботи:
   - принцип захисту – від доступу;
   - механізм захисту – автономний модуль;
   - об'єкт захисту – дані користувача;
   - середовище розробки – XCode;
   - середовище функціонування – ОС iOS.
4. Зміст розрахунково-пояснювальної записки: Вступ. Аналіз літературних джерел. Розробка адаптивного методу для автентифікації користувачів мобільних пристроїв. Розробка програмного застобу та експерементальне дослідження. Економічна частина. Висновки. Перелік використаних джерел. Додатки.
5. Перелік ілюстративного матеріалу.

Схема налаштування арифметичного методу (плакат, А4). Схема автентифікації за допомогою арифметичного методу (плакат, А4). Схема автентифікації за допомогою метода одноразових паролів (плакат, А4). Схема автентифікації за допомогою комбінованого метода (плакат, А4). Структура програмного засобу (плакат, А4).

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Лукічов В. В., к.т.н. ст. викл. каф. ЗІ		
2	Лукічов В. В., к.т.н. ст. викл. каф. ЗІ		
3	Лукічов В. В., к.т.н. ст. викл. каф. ЗІ		
4	Мацкевічус С. С., ст. викл. каф. ЕПВМ		

7. Дата видачі завдання _____ 2019 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	01.09.2019 – 04.09.2019	
2	Аналіз літературних джерел за напрямком магістерської кваліфікаційної роботи	05.09.2019 – 5.09.2019	
3	Науково-технічне обґрунтування	15.09.2019 – 26.09.2019	
4	Розробка технічного завдання	27.09.2019 – 30.09.2019	
5	Розробка рішень	30.09.2019 – 12.10.2019	
6	Практична реалізація, моделювання, експериментування, результати	12.10.2019 – 10.11.2019	
7	Розробка розділу економічного обґрунтування доцільності розробки	11.11.2019 – 17.11.2019	
8	Аналіз виконання ТЗ, висновки	19.11.2019 – 24.11.2019	
9	Оформлення пояснювальної записки	25.11.2019 – 30.11.2019	
10	Попередній захист та доопрацювання МКР	26.11.2019 – 01.12.2019	
11	Перевірка магістерської роботи на наявність плагіату	02.12.2019 – 10.12.2019	
12	Представлення МКР до захисту	11.12.2019 – 14.12.2019	
13	Захист МКР	16.12.2019 – 20.12.2019	

Студент _____ Вітович М. М.
(підпис)

Керівник роботи _____ Лукічов В. В.
(підпис)

АНОТАЦІЯ

Магістерська кваліфікаційна робота присвячена розробці програмного засобу для автентифікації користувачів в операційній системи iOS. Для успішної розробки програмного засобу проведено дослідження основних методів автентифікації користувачів операційної системи iOS, розроблено власні методи захисту: «арифметичний» метод, метод «одноразових паролів» та «комбінований» метод. Ці методи виправляють головні недоліки вже існуючих методів автентифікації. Під час роботи обґрунтовано вибір власних методів, розроблено ряд схем і алгоритмів, здійснено програму реалізацію. Засіб перевірено на коректність роботи, доведено ефективність здійснюваного захисту.

ABSTRACT

The master's thesis devoted to the development of software for authentication of users on the iOS operating system. For the successful development of the software, the basic methods of authentication of users of the iOS operating system were researched, their own methods of protection were developed: the "mathematical" method, the "one-time passwords" method and the "combined" method. These methods correct the major disadvantages of existing authentication methods. During work the choice of own methods is grounded, a number of schemes and algorithms is developed, the program of realization is realized. The tool is checked for correct work, the effectiveness of the protection is proved.

ЗМІСТ

ВСТУП

Наразі все більше людей не уявляє свого життя без смартфона або планшета під рукою. Смартфони є ключовим інструментом в багатьох сферах життя суспільства, від простого спілкування у соціальних мережах до управління персональними банківськими рахунками [1]. А однією з найпоширеніших та найпопулярніших операційних систем для смартфонів на сьогоднішній день є ОС iOS.

Слід визнати, що практично всі смартфони стали носіями конфіденційної інформації, важливих персональних або корпоративних даних. Разом з тим варто зазначити, що за допомогою смартфона легко отримати доступ до електронної пошти. Також за допомогою телефону власник може легко отримати доступ до своїх облікових записів в соціальних мережах таких як Facebook, Twiter, Instagram, а також до банківських карт. А з розвитком мобільного банкінгу, ця проблема стає ще важливішою. Адже якщо користувач проходить автентифікацію у фінансовому застосунку (приват24, монобанк та інші) то отримує доступ до всіх фінансових операцій.

Тому, всі ці дані повинні бути добре захищеними, але сам користувач не повинен хвилюватися про безпеку та захист цієї інформації. Тому актуально удосконалювати внутрішню безпеку операційної системи iOS, а саме процес автентифікації користувачів.

В той же час кожний користувач потребує різного рівня захисту, залежно від цінності збереженої інформації.

Оскільки в ОС iOS передбачено декілька варіантів для автентифікації користувачів, потрібно проаналізувати переваги та недоліки кожного з них і розробити адаптивний метод для автентифікації користувачів, щоб користувач в залежності від його потреб міг налаштувати потрібний йому рівень захисту своїх даних.

Мета роботи полягає в покращені методів захисту інформації та створення програмного засобу для удосконалення системи безпеки операційної системи iOS.

Об’єктом дослідження є процес автентифікації користувачів в операційній системі iOS.

Предметом дослідження є методи захисту конфіденційних даних користувачів мобільних пристроїв та дослідження існуючих методів автентифікації в ОС iOS.

Для досягнення мети необхідно виконати такі задачі:

– дослідити архітектуру та структуру безпеки ОС iOS;

– проаналізувати відомі методи автентифікації користувачів в ОС iOS та їх недоліки;

– розробити алгоритми адаптивного методу автентифікації;

– програмно реалізувати адаптивний метод автентифікації;

– реалізувати програмний засіб, оснований на адаптивних методах автентифікації;

– описати процес налаштування та встановлення програмного застосунку;

– протестувати розроблений програмний засіб з метою перевірки ефективності процесу автентифікації.

Наукова новизна одержаних результатів:

Удосконалено метод автентифікації користувачів мобільних пристроїв з ОС iOS, який відрізняється від існуючих аналогів тим, що, дані для автентифікації кожного разу є різними, що дозволяє значно ускладнити процес зламу або підглядання автентифікаційних даних.

1 АНАЛІЗ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1.1 Аналіз цілей та напрямків кіберзлочинності

Згідно зі статистикою на 2019 рік частка атак, що направлені на отримання інформації, продовжує зростати [1]. В 40% кіберінцидентів злочинці були націлені на отримання інформації, а в 38% – на фінансову вигоду [1].

В дослідженні компанії Positive Technologies під назвою «Ринок злочинних кіберпослуг» наведено аналіз пропозиції та попиту тіньового ринку на різноманітні дані [1]: персональні, платіжні, облікові тощо.

Також, як зазначено у дослідженні, 58% усіх пропозицій з продажу даних в «дарквебі» (тіньовий інтернет) – це облікові записи користувачів для доступу до різноманітних ресурсів, в тому числі до банківських застосунків [1]. Облікові записи продають як роздрібно по 12$, так і великими партіями по мільйону записів [2].

Тому після такої атаки, в якій була отримана інформація, доволі швидко може відбутися нова – на власників цих даних чи на компанію, облікові дані співробітників якої були скомпрометовані [1].

Серед інформації, яка найбільше приваблювала зловмисників, 35% складають персональні дані та 20% – облікові записи та парольна інформація для доступу до різноманітних сервісів і систем, в тому числі онлайн-банкам [1].

В 16% випадків були викрадені дані платіжних карток, вони найчастіше були отримані за допомогою шпигунського ПЗ або скомпрометованих сайтів [1].

Гістограма з частками скомпрометованої зловмисниками інформації, розподілена за приналежністю, наведена на рисунку 1.1.

Частка атак, націлених на інфраструктуру, в другому кварталі 2018 року склала 42%, а частка атак на веб-ресурси виросла у порівнянні з аналогічним періодом 2017 року і склала 34% проти 23% [1].

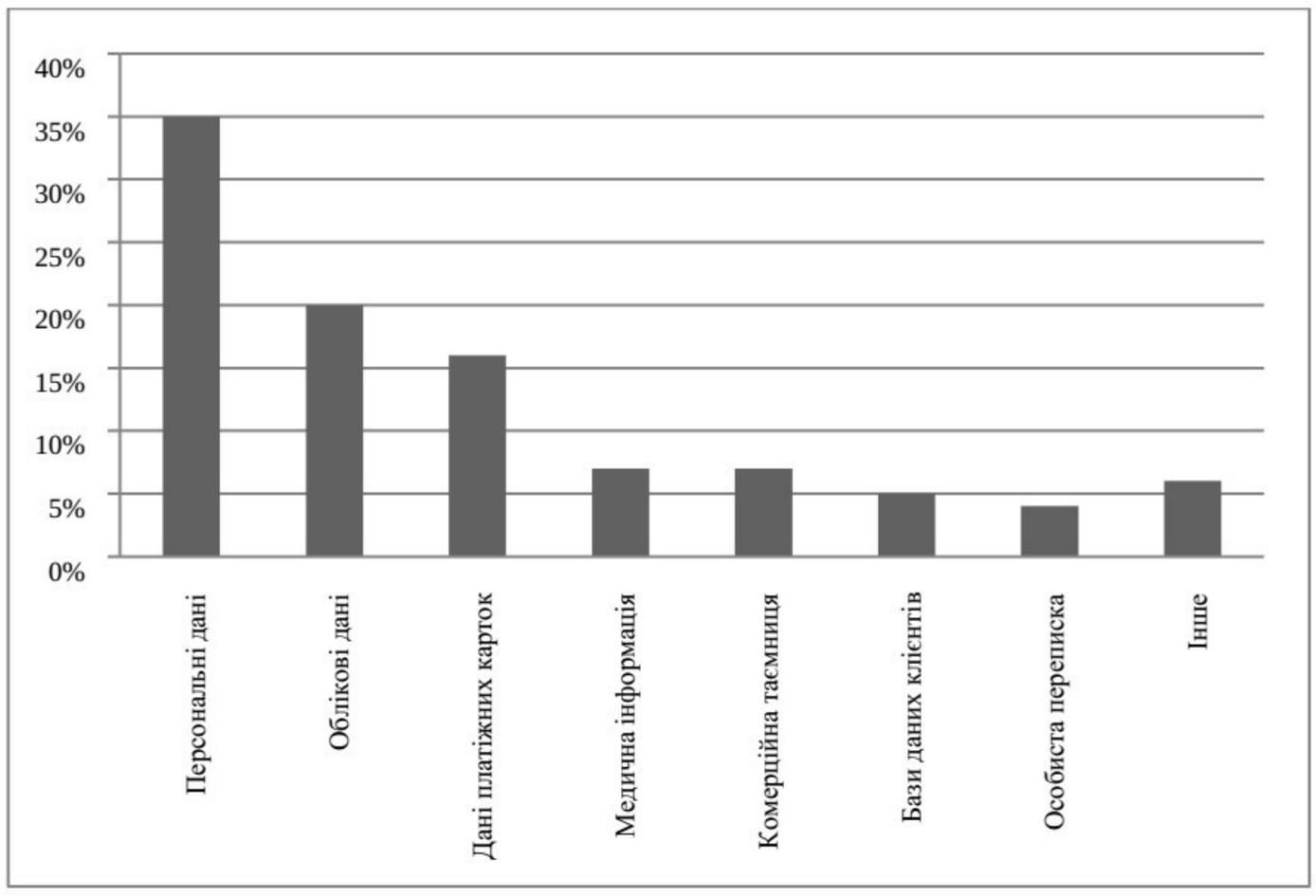


Рисунок 1.1 – Діаграма з частками скомпрометованої інформації

Крім того, в порівнянні з першим кварталом виросла частка атак на IoT-пристрої: це пов’язано з появою нових ботнетів, таких як, наприклад, PyRoMineIoT, Muhstik, Wicked Mirai [1].

Також у порівнянні з першим кварталом знизилася частка атак з використанням ЗПЗ (46% проти 62%) та на 14% виросла частка атак, основаних на підборі облікових даних [1].

На рисунку 1.2 представлена діаграма об’єктів кібератак [1].

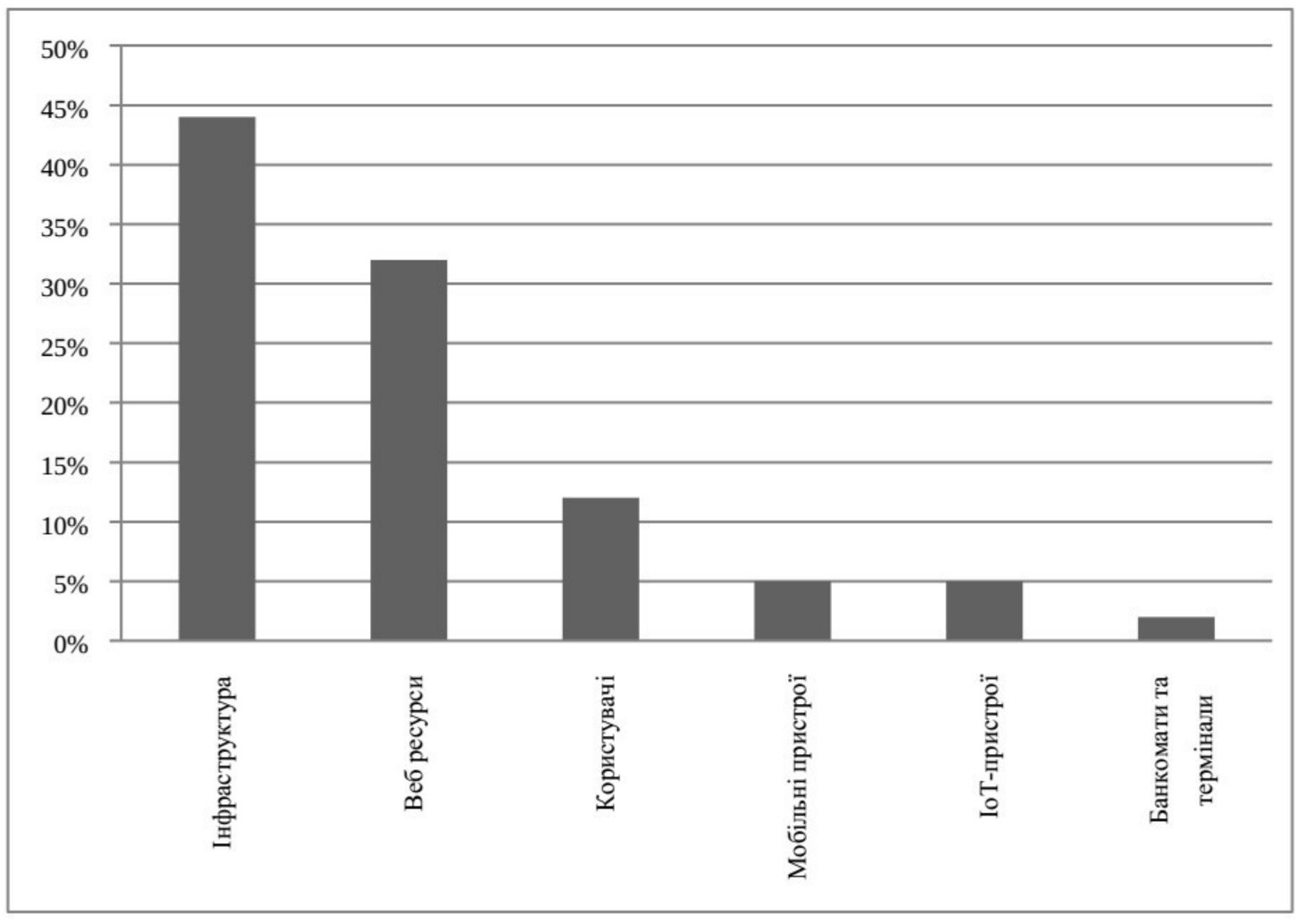


Рисунок 1.2 – Діаграма об’єктів кібератак

За підсумками другого кварталу 2018 року виділено такі тенденції [1]:

1. Кількість унікальних інцидентів продовжило зростати і на 46% перевищило показники аналогічного періоду 2017 року.

2. Переважали цілenаправлені атаки на конкретні організації та їх клієнтів, також на крипто валютні біржі. В ході цих атак зловмисники не тільки використовували ЗПЗ, але й шукали вразливості нульового дня, дізнавалися паролі адміністраторів за допомогою соціальної інженерії, отримували доступ до ресурсів контрагентів.

3. Відбулася велика кількість атак на крипто валюти, в результаті яких було викрадено більше 100 млн. дол. США.

4. Продовжила зростати частина кібератак, виконаних з ціллю отримання інформації. Зловмисники найбільше були зацікавлені в отриманні персональних та облікових даних, а також в даних банківських карт.

5. Приватні особи страждали від різноманітного ЗПЗ: більшу частину вони встановлювали самостійно через неуважність або незнання, хоча зустрічалися й такі методи атак коли нові смартфони продавалися в магазинах з уже встановленим в прошивку ЗПЗ.

За прогнозами досліджень [1] буде збережена тенденція до збільшення частки атак, направлених на викрадення даних. Багато компаній приділяють мало уваги захисту оброблюваної інформації, що робить її легкою здобиччю навіть для низько кваліфікованих зловмисників. Отримана інформація продається на тіньовому ринку та використовується для подальших атак на її власників [2].

1.2 Опис структури безпеки операційної системи iOS

Принцип безпеки закладений в саму основу ОС iOS. Оскільки будь-які дані користувача мають найвищий пріоритет, Apple спробували спроектувати операційну систему так, аби надати найвищий рівень безпеки для користувача інформації. В iPhone, iPad і iPod touch діє декілька рівнів безпеки. Низькорівневі апаратні функції захищають пристрій від шкідливого ПО, тоді як

високо рівневі функції iOS забезпечують безпечний доступ до призначених для користувача даних і корпоративної інформації, а також запобігають несанкціонованому використанню пристрою [2].

Завдяки тісному взаємозв'язку програмного і апаратного забезпечення в пристроях під управлінням iOS, кожен крок, починаючи від навантаження системи і закінчуючи установкою додатків, аналізується з точки зору безпеки та ефективності використання ресурсів. Цілісність системи безпеки безпосередньо залежить від цілісності і надійності ядра iOS – XNU [3]. На рисунку 2.1 схематично показана архітектура системи безпеки iOS.

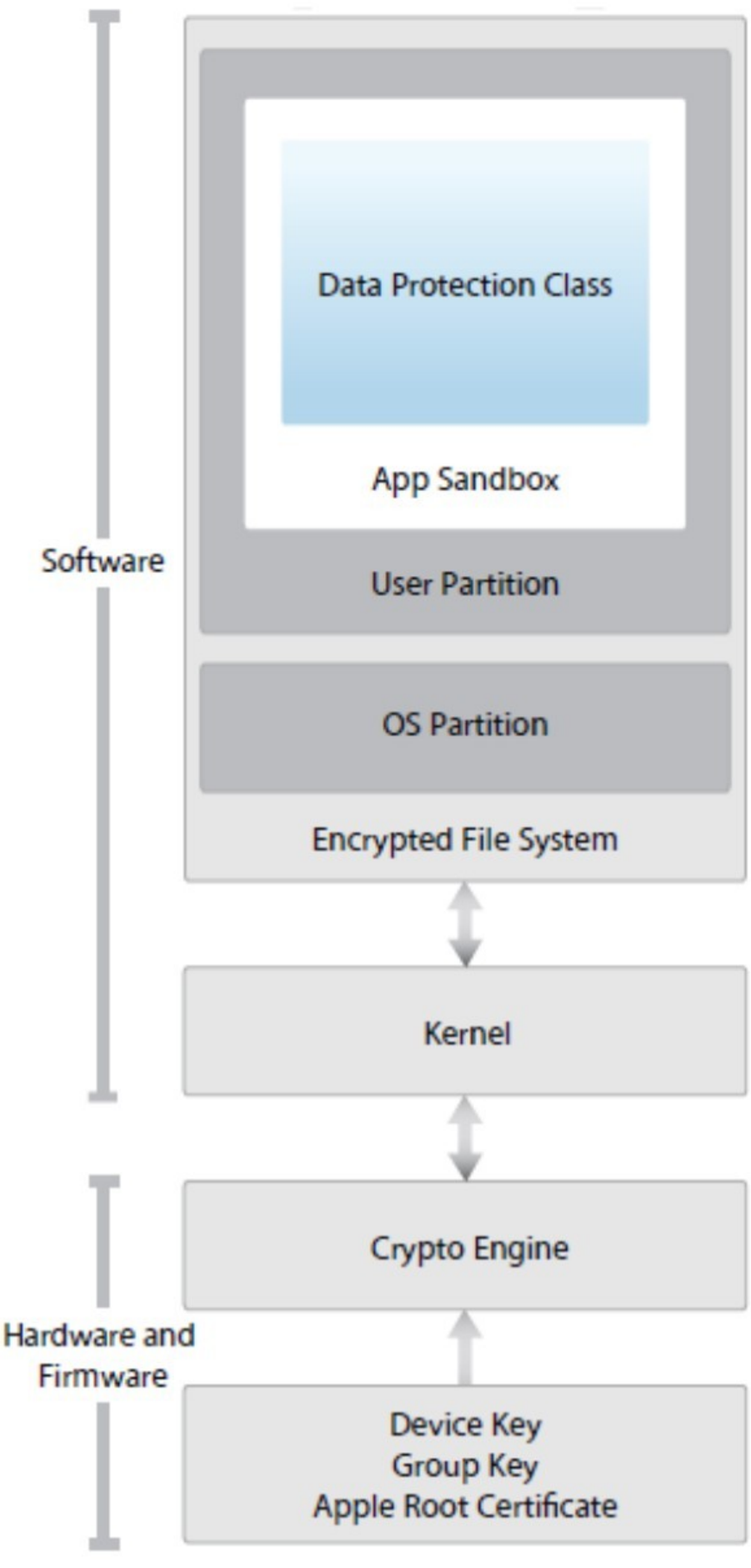


Рисунок 1.3 – Архітектура безпеки iOS

В iOS впроваджено так званий «ланцюжок довіреного завантаження». Кожен крок завантаження містить компоненти, які мають цифровий підпис Apple. На рисунку 2.2 зображена ланцюжок довіреного завантаження.

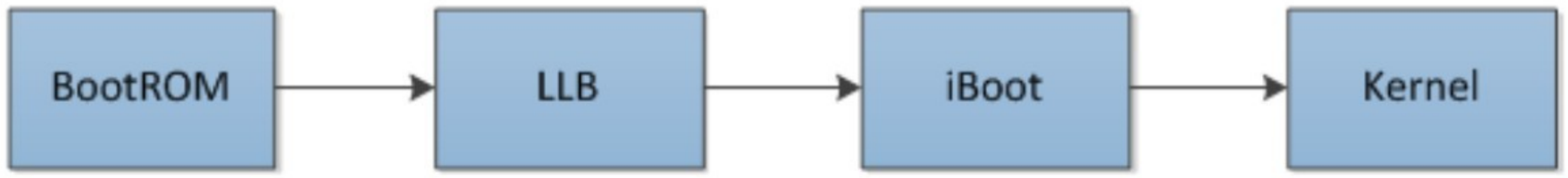


Рисунок 1.4 – Ланцюжок довіреного завантаження

При включенні пристрою процесор негайно починає виконувати код з read-only пам'яті, так званої BootROM. Код зашивається в чіп ще при виробництві, тому до коду є беззастережна довіра. BootROM також містить сертифікат кореневого центру сертифікації Apple. З допомогою сертифіката BootROM перевіряє цифровий підпис низькорівневого завантажувача (Low-Level Bootloader - LLB). LLB потім перевіряє підпис і запускає наступний завантажувач - iBoot. І, нарешті, iBoot перевіряє і запускає ядро [5].

Ланцюжок довіреної завантаження виконує дві функції. По-перше, вона гарантує, що низькорівневе ПЗ не піддалося ніяким несанкціонованим змінам; і по-друге, дозволяє iOS завантажуватися тільки на пристроях Apple.

Якщо на якомусь кроці в процесі не вдається перевірити чи запустити наступний крок, то завантаження припиняється, пристрій входить в режим відновлення (recovery mode) і на екрані відображається напис "Connect to iTunes". Якщо ж збій відбувається вже на першому кроці, тобто якщо BootROM не вдалося перевірити і завантажити LLB, то пристрій входить в режим DFU (Device Firmware Upgrade – оновлення прошивки пристрою). В обох випадках необхідно підключити пристрій до iTunes і повернутися до заводських налаштувань.

Apple періодично випускає оновлення безпеки, про які повідомляє користувача на самому пристрої або через iTunes. Apple робить все можливе,

щоб оновлення безпеки були встановлені на пристроях користувача якомога швидше.

Оновлення iOS можна встановити або за допомогою iTunes, або "по-повітрю" (через Wi-Fi). У першому випадку скачується і встановлюється повна копія iOS. При оновленні по Wi-Fi скачується лише необхідні "дельта-файли".

Як же працює механізм персоналізації системного ПО? Під час інсталяції або поновлення iOS iTunes або сам пристрій (в разі поновлення по Wi-Fi) підключається до сервера оновлення Apple (gs.apple.com) і посилає наступну інформацію:

– Список криптографічних параметрів кожного оновлення;

– Випадкове число;

– Унікальний ідентифікатор пристрою (ECID).

Отримавши дані від клієнта, сервер перевіряє, чи можлива установка такої версії поновлення на пристрій користувача. Якщо установка можлива, то сервер додає до параметрів ECID (саме ECID "персоналізує" пристрій), підписує їх (параметри і ECID) і відправляє назад користувачеві. Тільки після успішної перевірки користувач може продовжити установку.

Як уже згадувалося вище, BootROM містить сертифікат кореневого центру сертифікації Apple, тому ланцюжок довіреної завантаження зможе перевірити електронний підпис сервера оновлень.

Механізм персоналізації також гарантує, що не вдасться скопіювати стару версію iOS з одного пристрою на інший. Завдяки випадковому числу, порушник не зможе зберегти відповідь сервера, щоб в майбутньому використовувати його для зниження версії системи (атака повтором).

Проте, способи зниження версії системи існують. Але і тут є одна умова: для успішного зниження версії необхідно мати збережений SHSH-сертифікат прошивки тієї версії, на яку ви збираєтеся знижувати систему. SHSH-сертифікат - це унікальна цифровий підпис прошивки. SHSH-сертифікат для кожної версії прошивки відрізняється, саме тому, не маючи SHSH-сертифікат

старої прошивки, знизити версію системи не вийде. Крім того, необхідно внести зміни в файл hosts [5]. Це робиться для того, щоб iTunes взаємодіяв ні з сервером Apple, а з підробленим сервером, який завжди дозволяє встановлення старої прошивки. У кожній новій версії операційної системи Apple намагається виправляти вразливості попередньої системи.

Також особливої уваги потребує система резервного копіювання iOS. Шифрування локальних резервних копій в iOS 10.2 і новіших настільки стійке, що навіть топовий акселератор Nvidia GTX 1080 демонструє швидкість перебору не вище сотні паролів в секунду [8]. Таким чином, нескладний пароль всього з семи знаків завантажить один комп'ютер на найближче тисячоліття. Аж до виходу iOS 11 можна було призначити на резервну копію довгий випадковий пароль, який забезпечив би роботою всі комп'ютери світу до кінця віку. Але як показує практика, бочку меду можна зіпсувати все однією ложкою дьогтю. І ця ложка тут – код блокування. Якщо дізнатися цей код то можна отримати доступ до безлічі функції, серед яких:

– відключити Find my iPhone і блокування Activation Lock;

– включити двофакторну автентифікацію, якщо вона не була включена;

– змінити пароль від Apple ID / iCloud; з цим паролем отримати доступ до хмарним резервних копій, синхронізованим даними і паролів з хмарної зв'язки ключів iCloud Keychain, причому з усіх пристроїв, прив'язаних до цього облікового запису;

– скачати фотографії з iCloud Photo Library (згадали Celebgate);

– заблокувати або видалити дані з інших пристроїв, прив'язаних до даного Apple ID;

– скинути пароль на локальну резервну копію, підключити телефон до комп'ютера і витягти всі дані;

– витягти всі паролі з зв'язки ключів або переглянути їх на самому пристрої.

Зневірившись захистити свої пристрої від вразливостей зламу коду блокування, в Apple зважилися на відчайдушний крок, який викликав бурхливу неоднозначну реакцію з боку як журналістів, так і правоохоронних органів [7].

Про новий режим USB Restricted Mode, який то з'являвся, то зникав, то знову з'являвся в бета-версіях iOS 11, ми вже писали. Режим USB Restricted Mode повністю відключає будь-який обмін даними через вбудований в пристрій порт Lightning.

Єдине, що залишається доступним – зарядка; з точки зору комп'ютера, до якого буде підключений iPhone з активованим режимом USB Restricted Mode, пристрій нічим не буде відрізнятися від, наприклад, зовнішнього акумулятора.

Активується цей режим так:

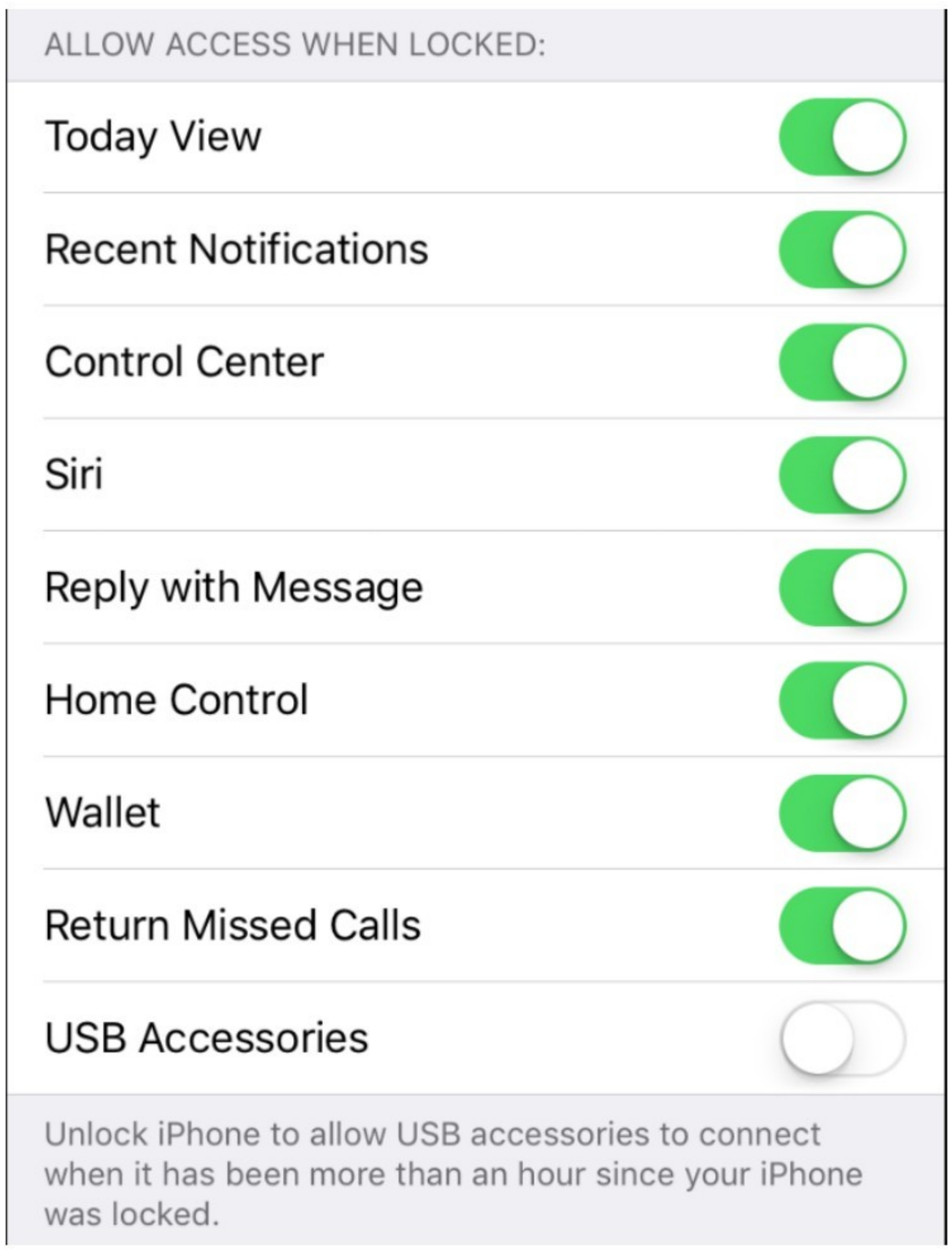


Рисунок 1.5 – Режим USB Restricted Mode активується при «вимкненому» положенні перемикача USB Accessories.

Комп'ютер при цьому не бачить нічого: недоступна навіть базова інформація про пристрій (така, як модель, серійний номер і версія iOS). На самому ж пристрої з'явиться ось таке спливаюче повідомлення:

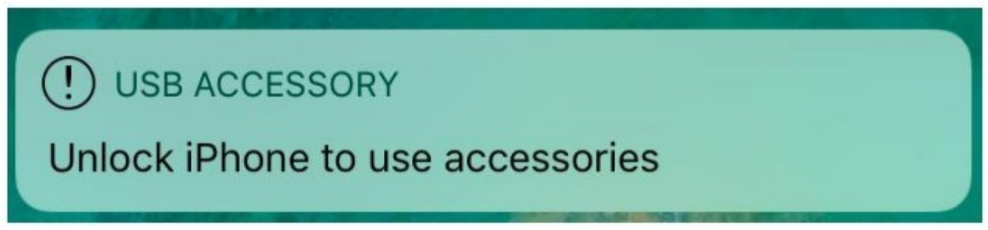


Рисунок 1.6 – Спливаюче повідомлення при ввімкненому режимі USB Restricted Mode

У поточній реалізації режим USB Restricted Mode з'явився в iOS далеко не відразу. Якщо в бета-версіях iOS 11.3.1 цей режим активувався через 6 днів з моменту останньої розблокування пристрою, то в iOS 11.4.2 USB порт відключався через годину.

Таким чином, якщо користувач не розблокував телефон протягом години (і не підключав його до довіреній пристрою), рішення Cellebrite або Gray Key для злому пароля не зможе працювати, так як не зможе підключитися до телефону [9].

Режим USB Restricted Mode досить надійний: телефон, як і раніше відмовиться спілкуватися з комп'ютером, навіть якщо його перезавантажити. Якщо перевести iPhone в режим Recovery або DFU, він стане доступний з комп'ютера – але перебір паролів в цих режимах неможливий.

В iOS 12.0 режим USB Restricted mode був додатково посилений. У попередньому релізі, в iOS 11.4.1, режим USB restricted mode автоматично відключав можливість передачі даних через порт Lightning через годину після блокування пристрою або його відключення від цифрового аксесуара.

Проте в iOS 11.4.2 залишалася можливість тривалістю в одну годину, протягом якого iPhone можна було підключити до перехідника (наприклад, Apple AV Adapter) або комп'ютера. Якщо зловмисник встигав це зробити, то iPhone можна було спокійно транспортувати в лабораторію, в якій згодом і здійснювалася атака.

Це вікно намагалися закрити в iOS 12.1. Тепер USB порт (і, відповідно, обмін даними) відключається відразу після блокування iPhone, але тільки в тому випадку, якщо користувач не підключав пристрій до комп'ютера або провідного аксесуару протягом трьох діб.

Втім, навіть якщо ви підключали iPhone до комп'ютера або перехідника протягом останніх трьох днів, USB порт все одно буде заблокований через годину після того, як ви в останній раз використовували пристрій.

Більш того, USB порт пристроїв в iOS 12 блокується і тоді, коли iPhone зажадає ввести код блокування для активації датчика відбитків або сканера особи, що в свою чергу відбувається за досить складним алгоритмом.

В цілому нововведення в iOS 12, безумовно, посилюють безпеку iPhone, не дозволяючи зловмисникові провести масштабну атаку на код блокування.

Також в iOS вбудований потужний механізм шифрування файлів на диску пристрою. Механізм дозволяє віддалено і швидко очистити файлову систему, захищає ваші дані при установці флеш-пам'яті в інший телефон.

Захист файлів реалізована через ієрархію криптографічних ключів. iOS присвоює кожному файлу 256 бітний AES ключ. Дані записуються на диск тільки в зашифрованому вигляді [11].

AES – симетричний алгоритм шифрування. Уряд США використовує AES з ключем 256 біт для зберігання секретних документів. Симетричність означає, що для шифрування і розшифровки даних використовується один і той самий ключ.

В iOS існує кілька класів захисту файлів. Кожен клас захисту має свій крипто-ключ. Файловий ключ шифрується ключем класу. Зашифрований файловий ключ зберігається в метаданих файлу. Метадані всіх файлів шифруються ключем файлової системи.

Ключ файлової системи створюється випадковим чином при першій установці iOS або очищенні пристрою. Коли користувач очищає пристрій опцією Erase all content and settings або віддалено через iCloud, ключ файлової

системи видаляється. Тепер всі файли недоступні, система не може розшифрувати метадані, в яких зберігається ключ файлу.

Ключ класу, в свою чергу, зашифрований унікальним ключем пристрою і паролем.

Така схема гнучка і продуктивна. Наприклад, щоб змінити клас захисту файлу, досить перешифрувати ключ файлу. А при зміні паролю досить перешифрувати ключ класу захисту [11]. Схему шифрування файлів показано на рисунку 1.7.

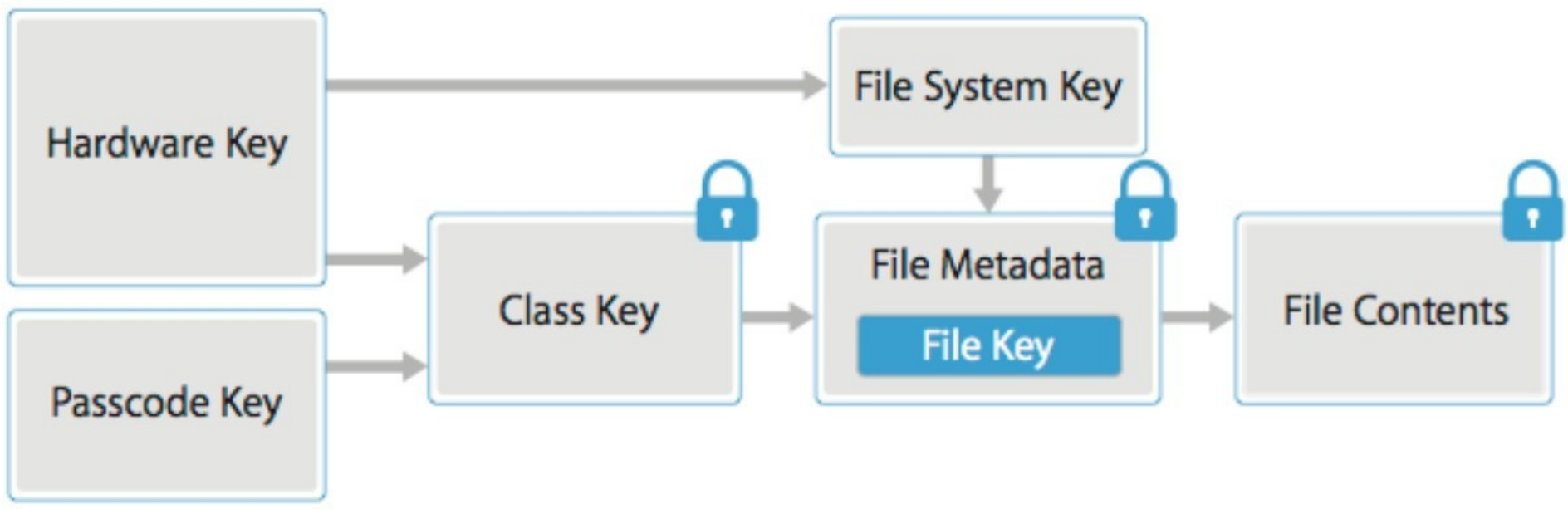


Рисунок 1.7 – Схема шифрування файлів в iOS

Постійні шифрування і розшифровування всіх файлів зповільнювали б роботу системи і зменшили б час роботи від батареї. Тому в кожному iPhone шифрування реалізовано на апаратному рівні.

На шляху між оперативною пам'яттю і флеш-картою вбудований чіп AES-256 шифрування. Процесор передає йому ключ файлу і дані прозоро і швидко шифруються і дешифруються при доступі до файлу.

Унікальний ключ пристрою (UID) і ключ групи пристроїв (GID) це 256 бітні AES ключі, вшиті в процесор при виробництві. Програмне забезпечення не має доступу до цих ключів, їм доступні лише результати шифрування / дешифрування цими ключами.

UID унікальний для кожного пристрою і не зберігається у Apple або її постачальників. GID збігається для всіх пристроїв з одним типом процесора (наприклад, для всіх пристроїв з A7 чіпом). GID використовується як

додатковий захист при установці і відновленні системи. Вшивання цих ключів на чіп дозволяє уникнути їх підміни та обходу [11].

UID дозволяє криптографічно прив'язати дані до конкретного пристрою. Наприклад, ієрархія ключів шифрування файлової системи включає UID.

Якщо переставити чіп пам'яті з одного пристрою на інший, файли будуть недоступні. Всі інші криптоключі належать системним генератором випадкових чисел..

Безпечне видалення ключів настільки ж важливо, як і їх створення. Це особливо складно на флеш-накопичувачах, де через зношування дані можуть дублюватися в декількох місцях. Тому iOS пристрої містять Effaceable Storage. Ця функція дозволяє стирати невеликі блоки даних з пристрою зберігання на найнижчому рівні.

1.3 Аналіз існуючих методів автентифікації в ОС iOS

На сьогоднішній день найпоширенішими методами автентифікації користувачів в ОС iOS є парольна та біометрична автентифікація, а також для доступу до різноманітних акаунтів можна використовувати двофакторну автентифікацію.

Найбільш розповсюдженою системою автентифікації є використання паролю у вигляді набору символів. Починаючи з операційної системи iOS 9.0 Apple впровадив 6-значні цифрові паролі. В Apple вважають, що стандартного чотиризначного PIN-коду для блокування пристрою буде недостатньо для забезпечення повної безпеки пристроїв, і вирішила впровадити в свою систему більш складні шестизначні паролі [12].

Якщо враховувати той факт, що всі сучасні мобільні пристрої Apple, крім Apple Watch, оснащені сенсором відбитків пальців, користувачі цих пристроїв навряд чи помітять великі відмінності від попередньої системи безпеки. Вводити для розблокування девайса, оснащеного технологією Touch ID, PIN-код не потрібно: достатньо просто прикласти палець до кнопки Home і iOS 9 сама дасть користувачеві доступ до системи.

На більш ж старих моделях інших варіантів, як розблокувати iOS, крім як ввести пароль, немає.

Збільшення кількості цифр в паролі цілком логічно, адже чим більше цифр у Вашого коду доступу, тим важче зловмисникові буде вгадати або дізнатися пароль.

Як стверджує Apple, установка шестизначного коду доступу передбачає 1 млн можливих комбінацій, а не 10000, як у випадку використання чотиризначного пароля. За допомогою спеціальних програм хакеру, щоб зламати чотиризначний код доступу, теоретично може знадобитися близько 111 годин, або 4,5 днів. У разі ж злому шестизначного пароля це займе аж 11000 годин, або 458 днів [6].

Також починаючи з iOS 10 з'явилася можливість встановлювати пароль з довільної кількості цифр та букв, що значно ускладню процес зламу такого паролю методом «грубої сили».

Яким би складним не був пароль, існує дві можливості його зламу (не враховуючи варіанти підглядування та викрадення з матеріальних носіїв): шляхом автоматичного підбору всіх можливих комбінацій знаків і використання програмної закладки, яка викрадає пароль із спеціальної області операційної системи.

Для боротьби із системами автоматичного підбору необхідно збільшувати кількість можливих комбінацій знаків, що призведе до збільшення необхідного часу для виконання операцій перебору всіх можливих варіантів. За цей час інформація застаріє і буде неактуальною або буде змінений пароль, і всю процедуру необхідно буде розпочати знову. Для боротьби з програмними закладками необхідно використовувати антивірусні програми.

Для всіх пристроїв iPhone і iPad існує як мінімум два працюючих незалежних рішення, що дозволяють зламати код блокування методом перебору. Ці рішення – Cellebrite і GrayKey – доступні виключно поліції і спецслужбам, але одного разу знайдена уразливість рано чи пізно опиниться в руках зловмисників [6].

Про рішення двох компаній офіційно відомо мало. Неофіційно ж ситуація досить цікава. Так, GrayKey прекрасно відпрацьовує на всіх пристроях під управлінням iOS 11.3.1 і більш ранніх версій, за умови, що зламувний телефон був хоча б раз розблоковано після включення або перезавантаження.

У таких випадках перебір йде дуже швидко, і код блокування з чотирьох цифр буде зламаний протягом максимум одного тижня.

Чи означає це, що код блокування, що складається з шести цифр, буде зламаний максимум через сто тижнів? Ні, не означає: через апаратні обмеження з такою швидкістю можна перебрати тільки 300 тисяч комбінацій з 10 мільйонів. Після цього перебір буде доступний лише в «повільному» режимі.

«Повільний» режим перебору дозволяє робити спроби з інтервалом в десять хвилин. Саме з такою швидкістю рішення компанії Grayshift буде перебирати паролі на пристроях, які були вимкнені або перезавантажувались (і жодного разу після цього не розблоковані). Більш того, починаючи з iOS 11.4 «повільний» режим перебору – єдине, що є і для раніше розблокованих пристроїв. Більше двох місяців на злом пароля з чотирьох цифр і майже дев'ятнадцять років для повного перебору простору паролів з шести цифр начебто цілком задовільні показники. Але не з точки зору Apple.

Процес введення пароля досить простий та швидкий, але не захищений від простого підглядання, або знімання на приховано розташовану відеокамеру.

Іншим недоліком системи парольної автентифікації є те, що пароль, насправді, дозволяє автентифікувати не конкретний суб’єкт, а лише зафіксувати відповідність автентифікатора суб’єкта його ідентифікатору, тобто пароль може беззастережно використовувати будь-який суб’єкт, незважаючи на те, яким чином він його отримав.

Функції символьної парольної автентифікації вбудовані в операційну систему iOS, тому побудова системи парольної автентифікації на їх основі не потребує додаткових витрат.

Іншим, більш популярним та надійнішим методом є біометрична автентифікація.

Біометрія – це метод автоматизованого розпізнавання людини за її унікальними фізіологічними або поведінковими характеристиками [4].

Останнім часом біометрична автентифікація стала стрімко розвиватись. Ряд потужних організацій на ринку інформаційних технологій (Microsoft, IBM, Novel, Compaq тощо) створили консорціум BioAPI, який має на меті зробити розпізнавання мови, обличчя і відбитків пальців базовими технологіями персональних комп'ютерів .

Взагалі існує багато різних варіантів біометричних методів. В iOS було реалазовані два: Touch ID і Face ID. Ідентифікація за відбитками пальців та лиця відповідно

Apple стверджує, що можливість збігу відбитків пальців – 1: 50000, а параметрів особи - 1: 1000000. Така низька статистика збігів доповнюється ще й тим, що біометричні дані не зберігаються на пристрої в чистому вигляді – вони перетворюються в математичну модель, яка не піддається розшифровці [12].

Паролі та відбитки (вірніше, їх математичні уявлення) знаходяться під захистом системи Secure Enclave, і тільки вона має до неї доступ. Secure Enclave ізольована від iOS, тому дані не будуть використані операційною системою та програмами, не потраплять на сервера Apple або в сховище iCloud.

Велика перевага біометричних методів автентифікації полягає в досить великій унікальності біометричних параметрів, які використовуються для автентифікації. Тому теоретично зламування біометричної системи автентифікації шляхом перебору можливих варіантів значення біометричного параметра надто складний, але з урахуванням недосконалості апаратно-програмних засобів біометричної автентифікації можливий.

Сама процедура біометричної автентифікації відносно проста (наприклад, прикласти палець, підставити під камеру, або пристрій для сканування обличчя або око) і не потребує будь-якого фізичного або психологічного напруження; немає потреби щось запам'ятовувати, періодично змінювати, або приховувати, чи постійно щось із собою носити.

З урахуванням того, що в біометричних системах інформація, яка використовується для автентифікації, незмінна, виникає можливість підміни біометричних параметрів (наприклад, виготовлення та використання силіконового пальця для дактилоскопічної системи).

Але ця процедура теж має певну вартість, яка може співвідноситись із вартістю самої системи автентифікації, і тому має сенс лише у випадку, коли вартість інформації, що захищається, суттєво вища вартості спуфінга [7]. Зі спуфінгом можливо боротися із застосуванням спеціальних технологій, наприклад, технології «живого пальця».

На відміну від будь-яких інших систем системи біометричної автентифікації дозволяють ідентифікувати саме суб’єкт автентифікації, а не його ідентифікатор, тому виключається можливість несанкціонованого застосування інформації, яка використовується для автентифікації, іншим суб’єктом. До того ж спрощується процедура контролю за потенційно небезпечними суб’єктами – у всіх інших випадках ідентифікатор та автентифікатор можна змінити.

Надійність збереження біометричних автентифікаторів (тобто біометричних параметрів суб’єкта) порівняно з символьними та графічними паролями досить велика, але системи біометричної автентифікації не захищені від випадків суттєвої зміни біометричних параметрів – пошкодження пальців, рук і інших частин тіла, які використовуються при автентифікації.

Важливий недолік біометричних систем автентифікації – неможливість одночасного зменшення рівня помилок першого та другого роду. Якість вирішення цієї проблеми пропорційна вартості систем. Також загальний недолік всіх біометричних методів – відносно висока вартість пристроїв для введення біометричної інформації.

Також для доступу до облікового запису iOS використовує двофакторну. При першому вході на новому пристрої вам буде потрібно надати два види інформації: ваш пароль і шестизначний цифровий код підтвердження, який автоматично відображається на довірених пристроях. Після введення коду

новий пристрій додається до довірених пристроїв. Наприклад, якщо у вас є пристрій iPhone, то при першому вході в обліковий запис на недавно придбаному комп'ютері Mac вам буде запропоновано ввести пароль і код підтвердження, який автоматично відобразиться на екрані вашого iPhone.

Оскільки для доступу до облікового запису при двофакторній ідентифікації недостатньо тільки знання пароля, безпека вашого Apple ID і збережених на серверах Apple даних істотно зростає.

Після виконання входу код підтвердження більше не буде запитуватися на цьому пристрої, поки не буде повністю виконано вихід, чи не будуть стерті всі дані на пристрої або поки не буде потрібно змінити пароль з міркувань безпеки. Після завершення входу через Інтернет можна вказати, що браузер є довіреним і наступного разу при виконанні входу з цього комп'ютера код підтвердження не буде запитуватися.

Довіреним пристроєм може бути iPhone, iPad або iPod touch з iOS 9 і пізніших версій, Apple Watch з watchOS 6 і пізніших версій або комп'ютер Mac з OS X El Capitan і пізніших версій, вхід в обліковий запис якого був виконаний з використанням двофакторної автентифікації. Це пристрій, для якого відома його приналежність вам і який можна використовувати для перевірки особистості шляхом відображення коду підтвердження Apple при вході з використанням іншого пристрою або браузера.

Код підтвердження – це тимчасовий код, що відправляється на довірений пристрій або довірений номер телефону при першому вході на пристрій або в браузер за допомогою ідентифікатора Apple ID.

Крім того, можна отримати код підтвердження в розділі «Налаштування» на довіреному пристрої. Код підтвердження генерується кожного разу новий та відрізняється від пароля до пристрою, що вводиться для розблокування iPhone, iPad і iPod touch.

2 РОЗРОБКА АДАПТИВНОГО МЕТОДУ ДЛЯ АВТЕНТИФІКАЦІЇ КОРИСТУВАЧІВ МОБІЛЬНИХ ПРИСТРОЇВ

Проаналізувавши переваги та недоліки основних відомих методів автентифікації користувачів в ОС iOS, доцільно розробити методи, які усуватимуть головні недоліки відомих методів. В даному розділі будуть описані декілька нових методів автентифікації: арифметичний, метод одноразових паролів, а також комбінований метод який дозволятиме адаптувати процес автентифікації для кожного користувача залежно від його потреб.

2.1 Побудова арифметичного методу автентифікації

Суть метода полягає в тому, що користувач повинен буде розв’язати деякий елементарний арифметичний приклад.

Але, не знаючи алгоритму, знайти правильну відповідь буде дуже складно. Даний метод дозволить збільшити час зламу автентифікаційних даних, не вимагаючи додаткових витрат апаратури, а також забезпечить захист від підглядання під час введення.

Нехай {a, b, c, d} – множина випадкових цілих чисел; {+, -, /, *} – множина можливих операцій над числами. Особливість арифметичного методу полягає в тому, що оператори будуть виконувати невластиві їм операції, а будь-які інші, які користувач обиратиме сам.

Також слід зазначити, що всі операції виконуватимуться послідовно, без будь-яких пріоритетів.

Наприклад, нехай + - виконуватиме операцію множення, – - операцію ділення, * - операцію додавання, / - операцію віднімання, тоді результат виразу (34-17*5/6+2) буде дорівнювати не 22, а 2. Адже насправді послідовно виконуватимуться такі дії 34/17+5-6*2.

Кількість чисел у прикладі користувач може задавати сам, мінімальна кількість операндів – 2, максимальна – 8. Операції, які буде виконувати

кожний з операторів, також задає сам користувач і може змінювати їх у вікні налаштувань.

Блок програмного засобу, що реалізовує даний метод автентифікації, передбачає два режими роботи з методом:

– налаштування параметрів автентифікації
– автентифікація.

Алгоритм роботи процесу налаштування представлений на рисунку 2.1.

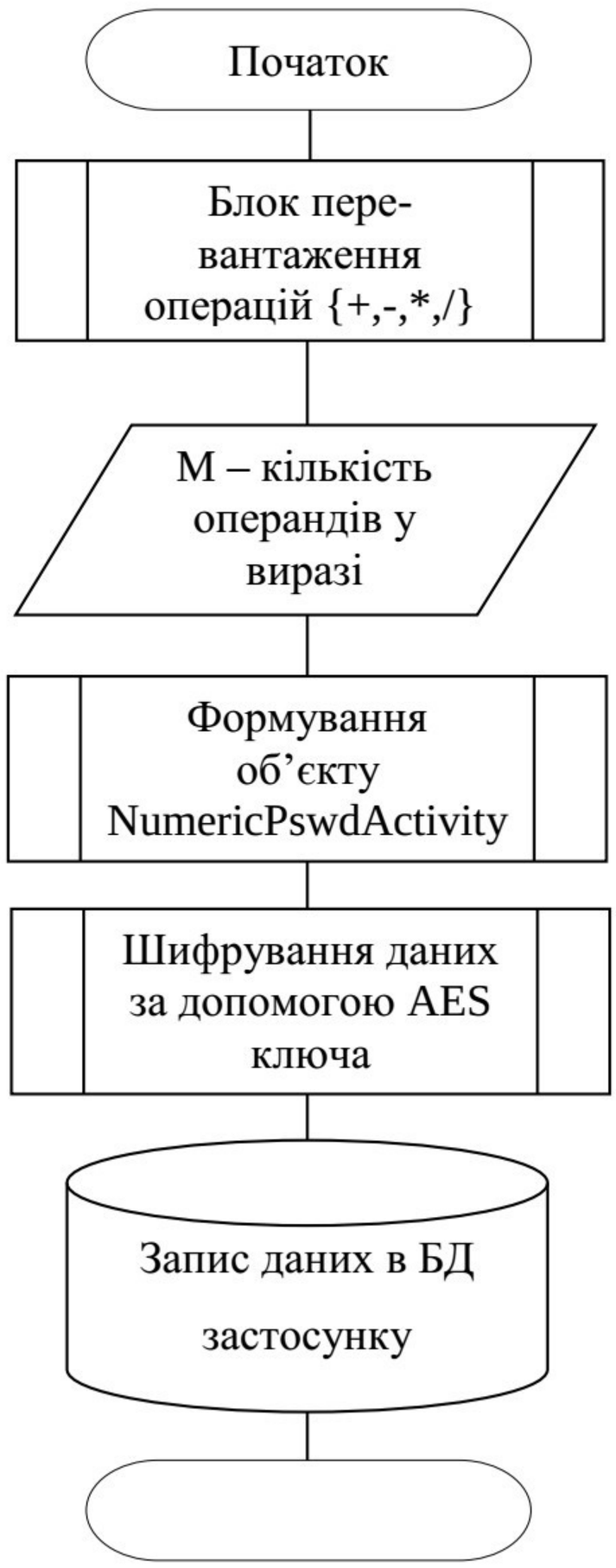


Рисунок 2.1 – Схема налаштування арифметичного методу

На рисунку 2.2 представлено схему автентифікації арифметичного методу.

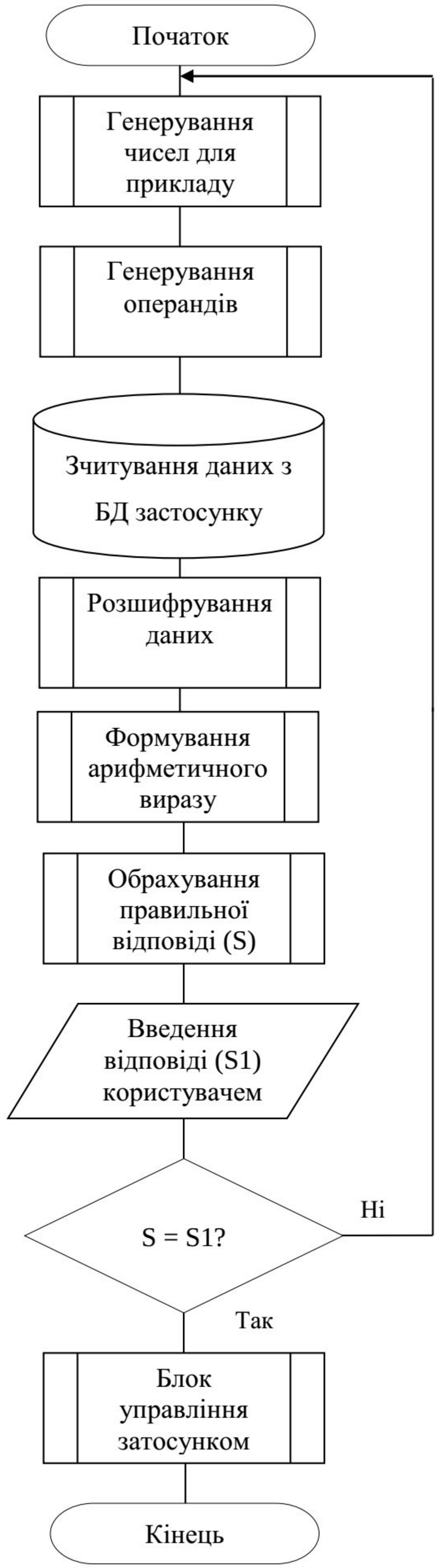


Рисунок 2.2 – Схема автентифікації за допомогою арифметичного методу

В режимі автентифікації виконуються такі дії:

– спочатку генеруються значення операндів для арифметичного виразу;

– далі генеруються операції, які прийматимуть участь у виразі;

– обраховується значення виразу згідно параметрів визначених в налаштуваннях методу;

– обраховане значення порівнюється зі значенням введеним користувачем;

– визначається результат автентифікації.

Надійність даного методу залежить від кількості задіяних операторів, тобто від кількості чисел в прикладі. В загальному надійність обраховується за формулою:

$$P = 4^{N-1},$$

де P – кількість можливих комбінацій, N – кількість чисел в прикладі.

Так, кількість можливих комбінацій для двох чисел в прикладі дорівнює лише 4, для 3 – 16, а от для 8 чисел це 65536 можливих комбінацій.

Отже, користувач сам вирішує, що йому важливіше: чи швидкість входу (адже чим більше чисел, тим складнішими і довшими будуть обрахунки), чи все ж таки надійність.

Також задля покращення безпеки використовується шифрування даних які зберігаються в базі даних. Для захисту даних використовується AES шифрування з довжиною ключа 128 біт.

AES Advanced Encryption Standard (оригінальна назва Rijndael) – стандарт розширеного шифрування, заснований на симетричному алгоритмі блочного шифрування, встановлений Національним інститутом стандартів і технологій США (NIST) в 2001 році [10]. Прийшов на зміну застарілому стандарту 3DES. Розмір блоку при шифруванні AES становить 128 біт, довжина ключа може становити 128, 192 або 256 біт. AES був прийнятий урядом США і тепер використовується у всьому світі. Він замінює стандарт шифрування даних (DES), який був опублікований в 1977 році. Алгоритм, описаний AES, являє

собою алгоритм з симетричним ключем, тобто той же ключ використовується для шифрування і дешифрування даних [11].

Алгоритм AES перетворює блок довжиною 128 бітів в інший блок тієї ж довжини. Для перетворення застосовується розклад ключів w отримується з ключа. 128-бітний блок в AES представляється у вигляді матриці 4 × Nb. Стандарт допускає тільки одне значення Nb = 4, тому довжина блоку завжди 128 біт, хоча алгоритм може працювати з будь-яким Nb. Довжина ключа дорівнює 4Nk байт. Алгоритм шифрування блоку складається з Nr раундів - застосувань однієї і тієї ж групи перетворень до 128-бітного блоку даних. В таблиці 2.1 наведено комбінації цих трьох параметрів, які допускає стандарт:

Таблиця 2.1 – Комбінації параметрів які допускає стандарт AES

	N_k	N_b	N_r
AES-128	4	4	10
AES-256	6	4	12
AES-256	8	4	14

Для шифрування тексту AES застосовує не пароль або геш від пароля, а так званий «розклад ключів», який отримується з ключа. Цей розклад можна уявити як Nr + 1 матриць розміру 4 × Nb. Алгоритм шифрування робить Nr + 1 кроків і на кожному кроці він, крім інших дій, бере одну матрицю 4 × Nb з «розкладу» і поелементно додає її до блоку даних.

Тому для шифрування даних для автентифікації було вибрано саме AES шифрування з довжиною ключа 128 біт.

Даний метод автентифікації вирішує проблему підглядання пароля під час введення, притаманну символьним і графічним паролям, внаслідок того, що числа і оператори кожного разу випадкові і результат завжди буде різним. Перевагою даного метода є також те, що він не потребує додаткових пристроїв,

а, отже, вирішує головний недолік біометричних методів – високі витрати на реалізацію.

2.2 Розробка методу одноразового паролю для автентифікації

Оскільки одним з найголовніших недоліків більшості методів автентифікації є можливість підглядання під час введення, тому доцільно розробити такий метод автентифікації за якого дані для автентифікації будуть завжди різними.

Суть методу одноразових паролів полягає в тому що при автентифікації буде генеруватись випадковий буквено-символьний пароль з 4-8 символів, який відправлятиметься користувачеві на пошту. Для успішної автентифікації користувач повинен ввести цей пароль. При наступній процедурі автентифікації буде згенеровано новий пароль. Кількість символів в паролі користувач зможе налаштовувати сам.

Отже навіть якщо зловмисник зможе побачити ваш пароль, то при спробі самому автентифікуватися у нього нічого не вийде.

Процедуру автентифікації зможуть пройти лише ті хто має доступ до вашої пошти.

Даний метод підходить для тих для кого захист даних та конфіденційність є важливішим ніж швидкість та доступність. Оскільки для автентифікації необхідно обов’язково мати доступ до пошти, то пройти процедуру автентифікації можна лише за наявності іншого пристрою. Це може бути інший смартфон, комп’ютер, ноутбук, планшет або смарт-годинник.

Ще кілька років тому даний метод передбачав би справді довгу та складну процедуру, але зараз у більшості людей практично завжди є доступ до кількох пристроїв. На сьогоднішній день у багатьох людей є смарт-годинник, який зазвичай зв’язаний з його поштою, а тому повідомлення з паролем він зможе отримати та ввести майже миттєво та без особливих труднощів.

Алгоритм роботи даного методу автентифікації представлений на рисунку 2.3.

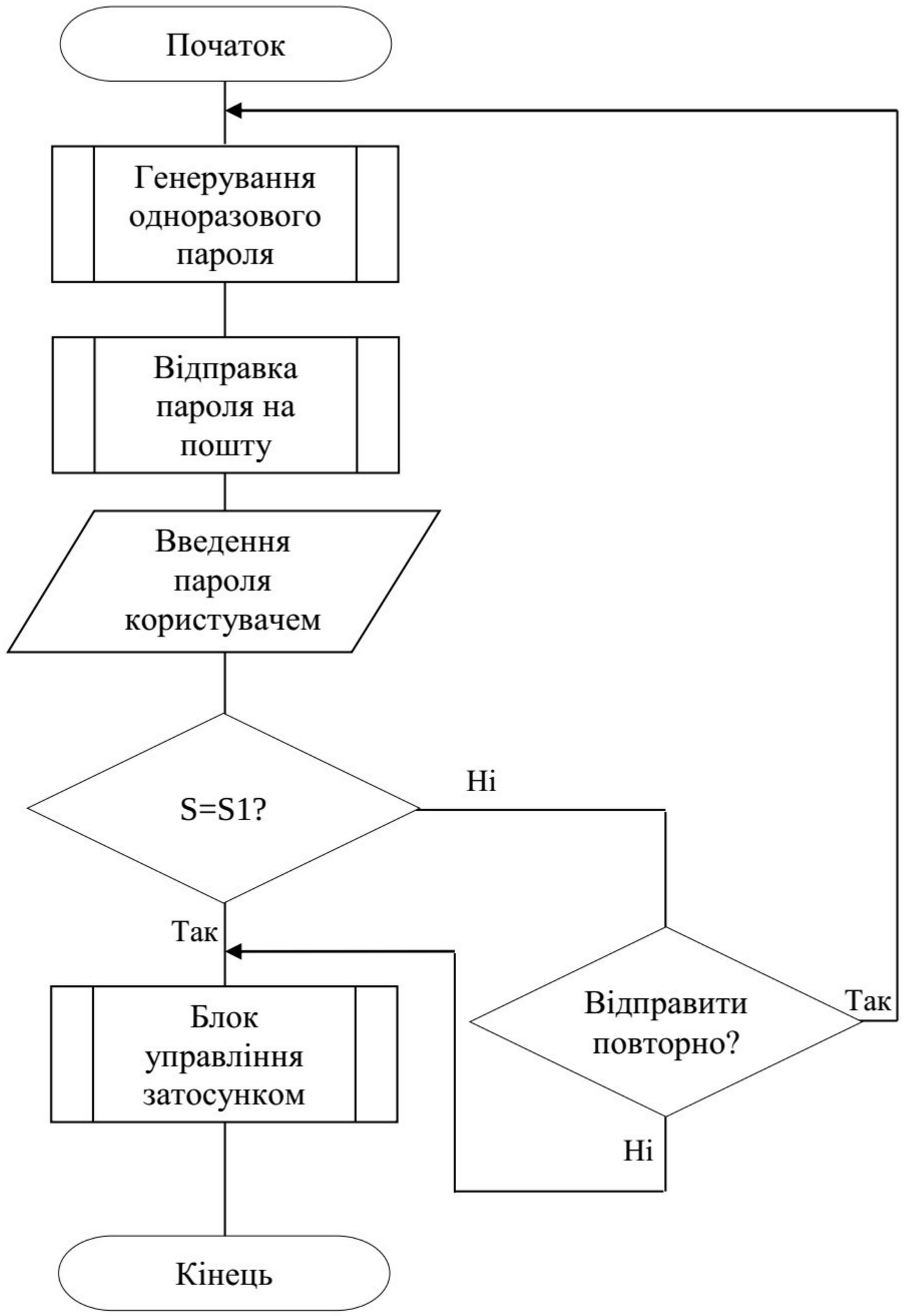


Рисунок 2.3 – Схема автентифікації за допомогою метода одноразових паролів

Алгоритм методу одноразових паролів передбачає виконання таких дій:

1. На першому етапі генерується одноразовий пароль довжиною в 4-8 символів;

2. Далі згенерований пароль, відправляється, на вказану користувачем при реєстрації, пошту;

3. Користувач вводить отримай пароль в поле для введення;

4. Введений користувачем пароль (S) порівнюється з правильним паролем (S1):

$$S = S1?$$

5. Якщо паролі співпадають, тобто S = S1, то процес автентифікації завершується і користувач отримує доступ до управління додатком;

6. Якщо ж пароль введений користувачем не співпадає з правильним, тобто S ≠ S1, то користувач має можливість отримати новий одноразовий пароль на пошту;

7. Якщо користувач вибирає режим «повторно отримати пароль», то алгоритм повторюється з пункту 1.

8. Якщо користувач не бажає повторно отримати пароль, то процес автентифікації завершується і користувач не отримує доступу до управління додатком.

Отже, даний метод автентифікації унеможливлює одну з головних проблем символьних і графічних паролів – підглядання пароля під час введення, за рахунок того що пароль кожного разу змінюється і не маючи доступу до вашої пошти ніхто не зможе пройти автентифікацію. Перевагою даного метода є також те, що він, як і попередній метод, не потребує додаткових пристроїв, а, отже, вирішує один з головних недоліків біометричних методів – високі витрати на реалізацію.

2.3 Розробка комбінованого методу для автентифікації

Даний метод передбачає можливість комбінації декількох методів для автентифікації користувачів.

У даному режимі, користувачу доступні 4 методи для автентифікації:

– пароль;

– біометрична автентифікація;

– арифметичний метод;

– одноразовий пароль.

Алгоритм роботи даного методу автентифікації представлений на рисунку 2.4, де S1 – парольна автентифікація, S2 – біометрична автентифікація, S3 – арифметичний метод , S4 – метод одноразових паролів.

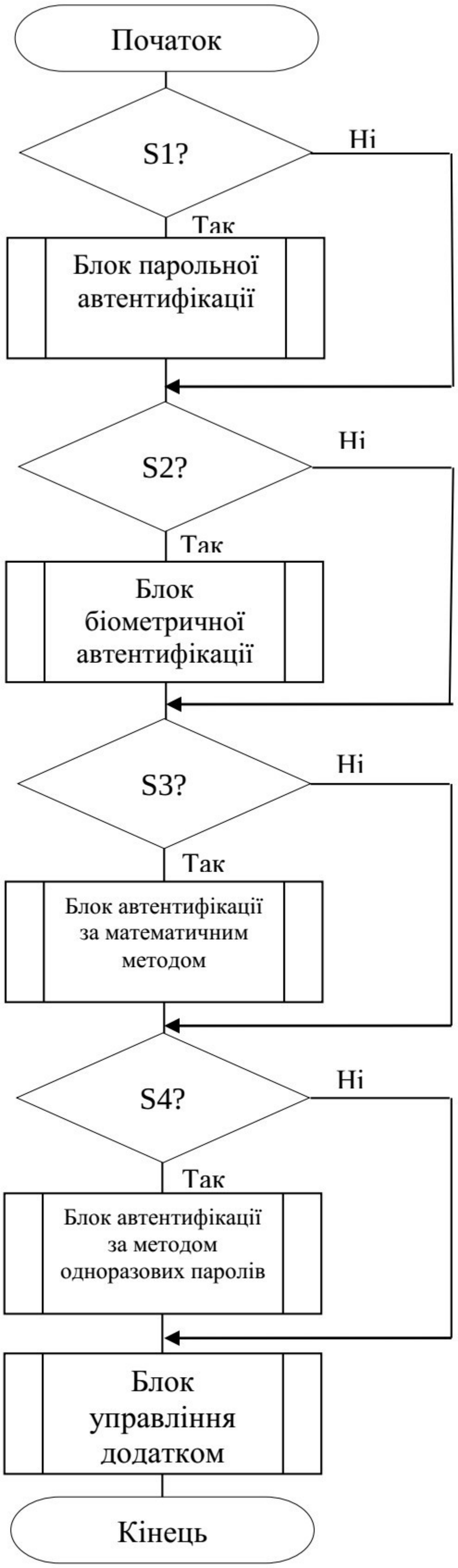


Рисунок 2.4 – Схема автентифікації за допомогою комбінованого метода

В процесі автентифікації відбувається покрокова перевірка кожного методу. Якщо метод використовується то викликається блок автентифікації за даним методом.

Алгоритм роботи автентифікації за арифметичним методом та методом одноразових паролів було показано на рисунках 2.2 та 2.3 відповідно.

Алгоритм роботи парольної автентифікації наведено на рисунку 2.5.

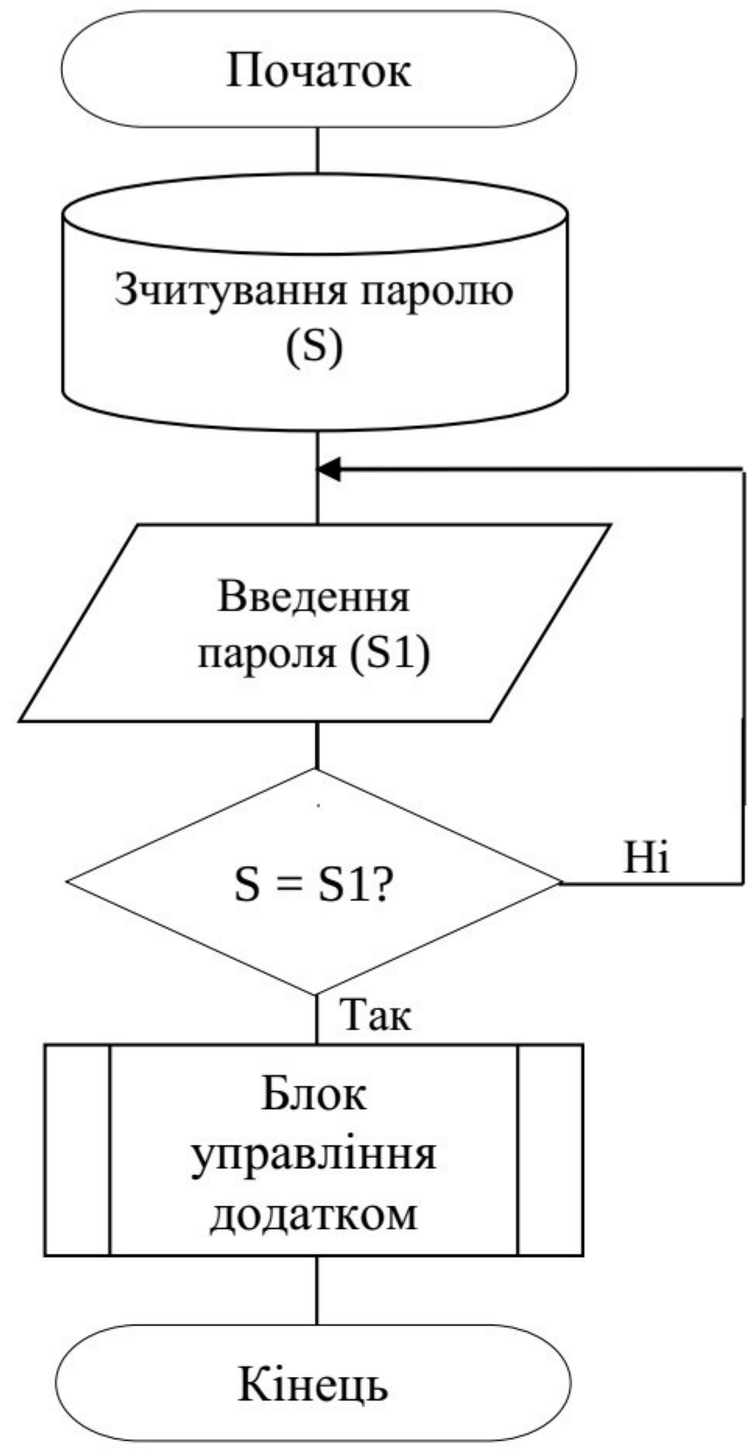


Рисунок 2.5 – Схема автентифікації за допомогою пароля

Алгоритм автентифікації за біометричним методом практично ідентичний з парольним методом, тільки замість пароля, скануються та порівнюються біометричні дані користувача.

Отже, даний метод є комбінацією існуючих та власних розроблених методів автентифікації. Він є доволі гнучким та дозволяє адаптувати під вимоги та потреби конкретного користувача. Надійність даного методу залежить від кількості методів, які вибирає сам користувач. Головною перевагою даного метода є адаптивність та надійність, а недоліком є складність та тривалість автентифікації.

3 РОЗРОБКА ПРОГРАМНОГО ЗАСТОБУ ТА ЕКСПЕРЕМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ

Програмний засіб складатиметься з модуля автентифікації за одним з трьох методів, та модуля реєстрації та вибору методів.

Структура програмного засобу показана на рисунку 3.1.

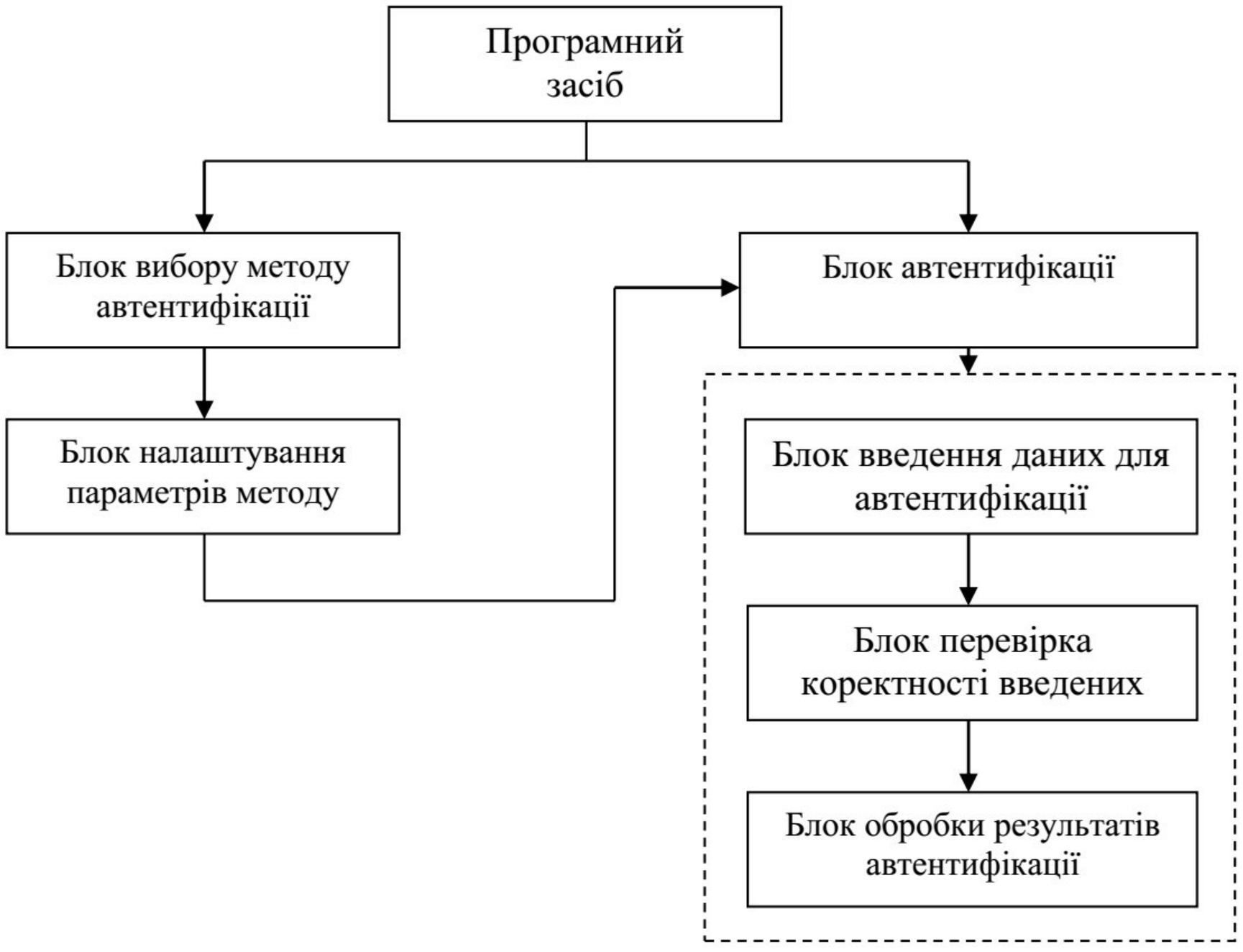


Рисунок 3.1 – Структура програмного засобу

Коли користувач вперше відкриває застосунок, йому потрібно пройти процес реєстрації та вибрати один з трьох методів для автентифікації. Вибравши один з методів, слід налаштувати його параметри входу. При наступному використання застосунку спершу слід буде автентифікуватися за вибраним методом, якщо введені дані будуть коректними, тоді користувач отримує доступ до додатку.

Розробка програмного засобу також буде виконуватись блоками, що є зручним при будь-яких модифікаціях та змінах програми.

3.1 Обґрунтування вибору програмних засобів для реалізації

З огляду на те, що програмний засіб призначений для автентифікації користувачів мобільних пристроїв з ОС iOS, для реалізації програми найкраще використати мову програмування Swift, а середовищі програмування xCode.

Swift є однією з найпопулярніших мов програмування, що використовуються розробниками програмного забезпечення на сьогоднішній день. Ядро мови використовується при розробці iOS-додатків.

Традиційно основною мовою програмування під iOS і MacOS був Objective-C, проте 2 червня 2014 року на конференції розробників Apple WWDC 2014 була представлена нова і більш зручна мова програмування – Swift [13]. У порівнянні з Objective-C Swift володіє наступними особливостями:

- swift є об'єктно-орієнтованою мовою програмування;
- простота, ясність та чіткий синтаксис;
- сувора типізація – кожна змінна має певний тип;
- автоматичне управління пам'яттю;

Однак при цьому Swift повністю сумісний з раніше написаними прикладними інтерфейсами Cocoa API, для яких використовувалися C і Objective-C.

Програмний засіб, що розробляється в рамках дипломної роботи, спроектовано на засадах ООП – об’єктно-орієнтованого програмування [14]. ООП – це парадигма програмування, заснована на поданні програми у вигляді сукупності взаємодіючих об'єктів, кожен з яких є екземпляром певного класу, а класи є членами певної ієрархії наслідування. Основні концепції, що характеризують ООП: успадкування, модульність, поліморфізм та інкапсуляція.

Оскільки дуже важливу роль відіграють розмір, швидкість виконання, наявність дружнього графічного інтерфейсу, доцільно обрати саме Swift для реалізації поставленої задачі.

xCode – інтегроване середовище розробки (IDE) компанії Apple, яке надає розробникам інструменти для створення додатків під iPhone, iPad, Mac, Apple

Watch і Apple TV. xCode запускається тільки на комп'ютерах з OS X (iMac, Macbook і Mac Mini).

У xCode IDE використовується схема поділу даних програми Model-View-Controller (Модель-Представлення-Контролер або MVC) для сегментації кожного шару додатка. Так простіше вносити зміни в код. Наприклад, шар UI розділений інструментами, такими як новий Interface Builder, з його допомогою можна поміщати на екран засоби візуального контролю. Auto Layout дозволяє динамічно управляти презентацією об'єктів для екранів різних розмірів; за допомогою Storyboard зручно розташовуються екрани додатка; режим Preview швидко покаже, як виглядають екрани додатка. Жоден з цих інструментів не зачіпає програмний код, який ви створюєте.

Отже, для реалізації дипломної роботи буде доцільно використати саме ці програмні засоби.

3.2 Програмна реалізація та тестування блоку реєстрації

Розробка програмного застосунку побудована на контролерах (View controller). Контролер прийнято сприймати як екран. Найпростіші додатки складаються з одного контролера. Більш складні програми можуть мати декілька екранів, тобто вони складаються з декількох контролерів, якими треба вміти управляти і які можуть взаємодіяти між собою [15].

Щоб створити контролер, його слід успадкувати від батьківського UIViewController. Життєвий цикл контролера починається з виклику функції viewWillAppear() та viewDidAppear(). В результаті отримуємо порожній екран. Сенсу від такого екрану ніякого. Тому в активність додаємо компоненти, фрагменти за допомогою розмітки.

Головний контролер застосунку називається MainViewController. На ньому користувач повинен вибрати режим роботи застосунку. Контролер містить дві кнопки – «Вхід» та «Реєстрація» (рис. 3.1).

Рисунок 3.1 – Вигляд головного контролера застосунку

При першому запуску додатку слід вибирати режим реєстрації. При виборі даного режиму викликається наступний контролер – RegistrationViewController(), за допомогою виклику наступної функції:

```
@IBAction func registButtonClick(_ sender: UIButton) {
     let vc = RegistrationViewController.instance(.main)
     navigationController?.pushViewController(vc, animated: true)
}
```

Екран контролера реєстрації містить кнопку «Далі» та три поля:

1. Поле для введення імені.
2. Поле для введення фамілії;
3. Поле для введення email.

Для реалізації даного контролера слід використати елементи UITableView() та UITableViewCell(). Для створення цих елементів слід імпортувати стандартну бібліотеку UIKit. Імпортувати дану бібліотеку можна за допомогою команди:

```
import UIKit
```

Вигляд контролера реєстрації показано на рисунку 3.2.

Рисунок 3.2 – Вигляд контролера реєстрації користувача

Слід передбачити, що поля не можуть бути порожніми, у випадку якщо користувач не заповнить їх, потрібно виводити діалог з повідомленням (рис. 3.3).

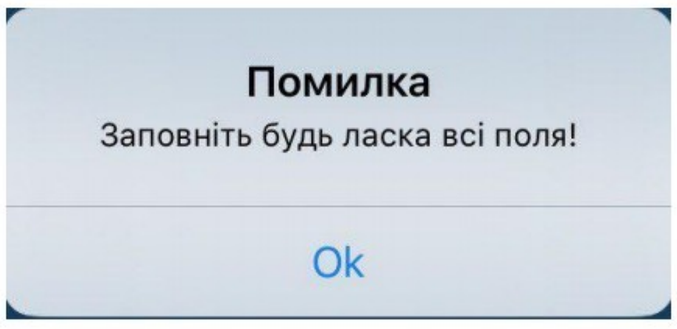


Рисунок 3.3 – Вигляд діалогового вікна з повідомленням

Також варто врахувати те, що в полі email користувач повинен ввести коректні дані, для цього слід добавити перевірку на валідність email за допомогою регулярного виразу:

```
func isValidEmail(_ text: String) -> Bool {
if text.range(of:"^[a-z_0-9]+(\\.[a-z_0-9\\-\\+]+)*@[a-z]+\\.[a-z]{2,6}$",
options: .regularExpression) == nil { return true } else { return false }
}
```

У разі не коректного введення даних у поле email, користувач отримає відповідне повідомлення (рис. 3.4).

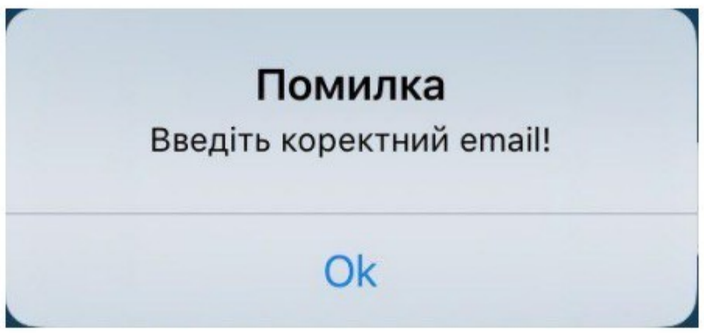


Рисунок 3.4 – Вигляд діалогового вікна про помилку email

Коли користувач успішно проходить реєстрацію, далі слід вибрати бажаний метод автентифікації. Користувачеві доступно три методи автентифікації, кожний з яких більш детально описано у розділі 2 даної магістерської роботи.

Вигляд контролера з вибором метода автентифікації показано на рисунку 3.5.

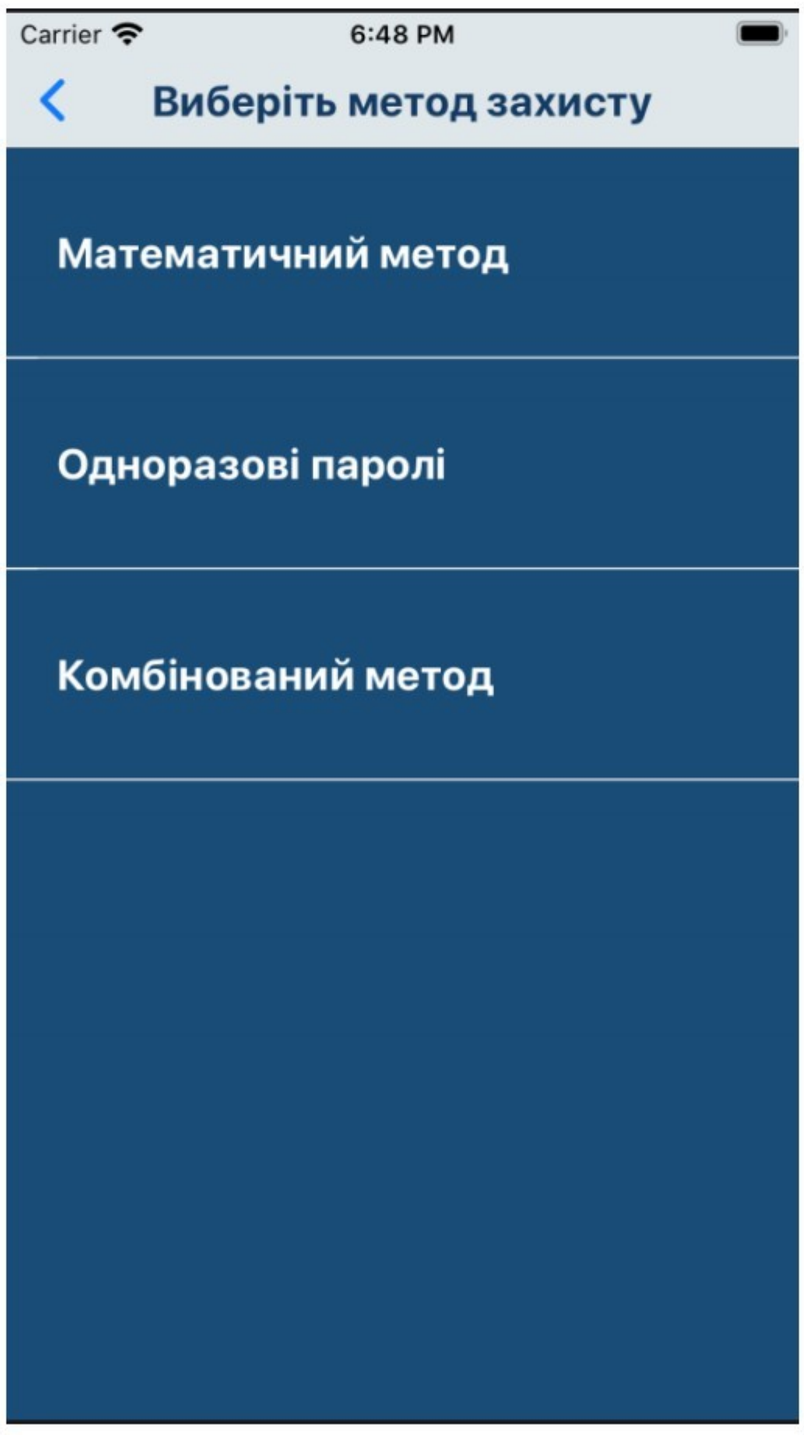


Рисунок 3.5 – Вигляд контролера вибору методу захисту

Для того щоб вибрати один з методів слід просто натиснути на нього.

Вибравши бажаний метод користувач потрапляє на контролер налаштування вибраного методу.

Контролер налаштування арифметичного методу містить чотири параметри:

1. Налаштування дій для знака «+»;
2. Налаштування дій для знака «–»;
3. Налаштування дій для знака «×»;
4. Налаштування дій для знака «÷»;

Вигляд екрану налаштування параметрів арифметичного методу автентифікації показано на рисунку 3.6.

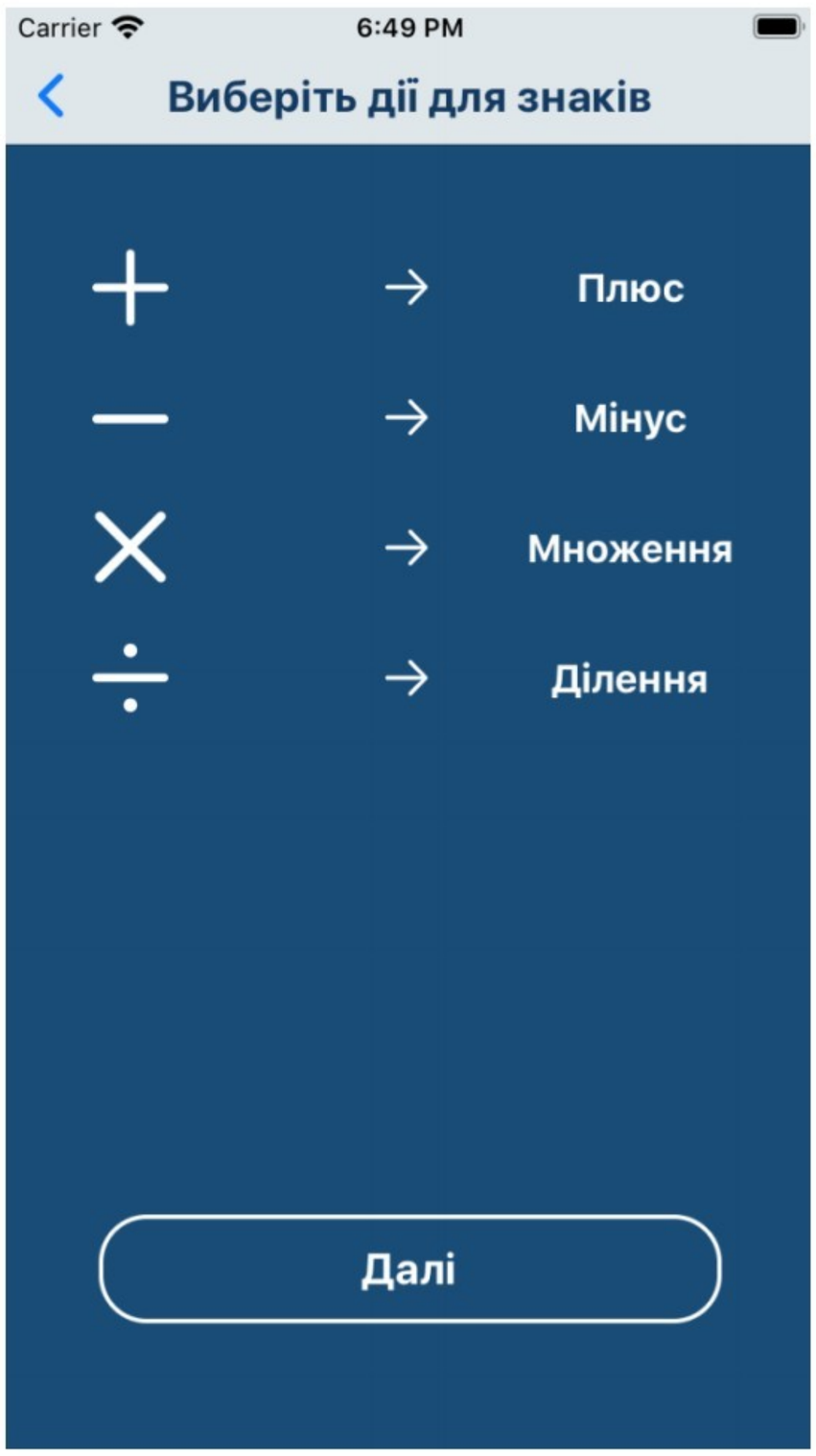


Рисунок 3.6 – Вигляд екрану налаштування арифметичного методу автентифікації

Для виклику меню налаштування кожного з знаків, потрібно натиснути на кнопку розташовану навпроти самого символу.

При натисканні на кнопку налаштування викликається функція:

```
@IBAction func plusButtonClick(_ sender: UIButton) {
    displayActionSheet(button: sender)
}
```

Меню вибору дії для конкретного знаку містить 5 опцій:

– множення;

– ділення;

– віднімання;

– додавання;

– відміна.

Вигляд вікна налаштування конкретного знаку показано на рисунку 3.7.

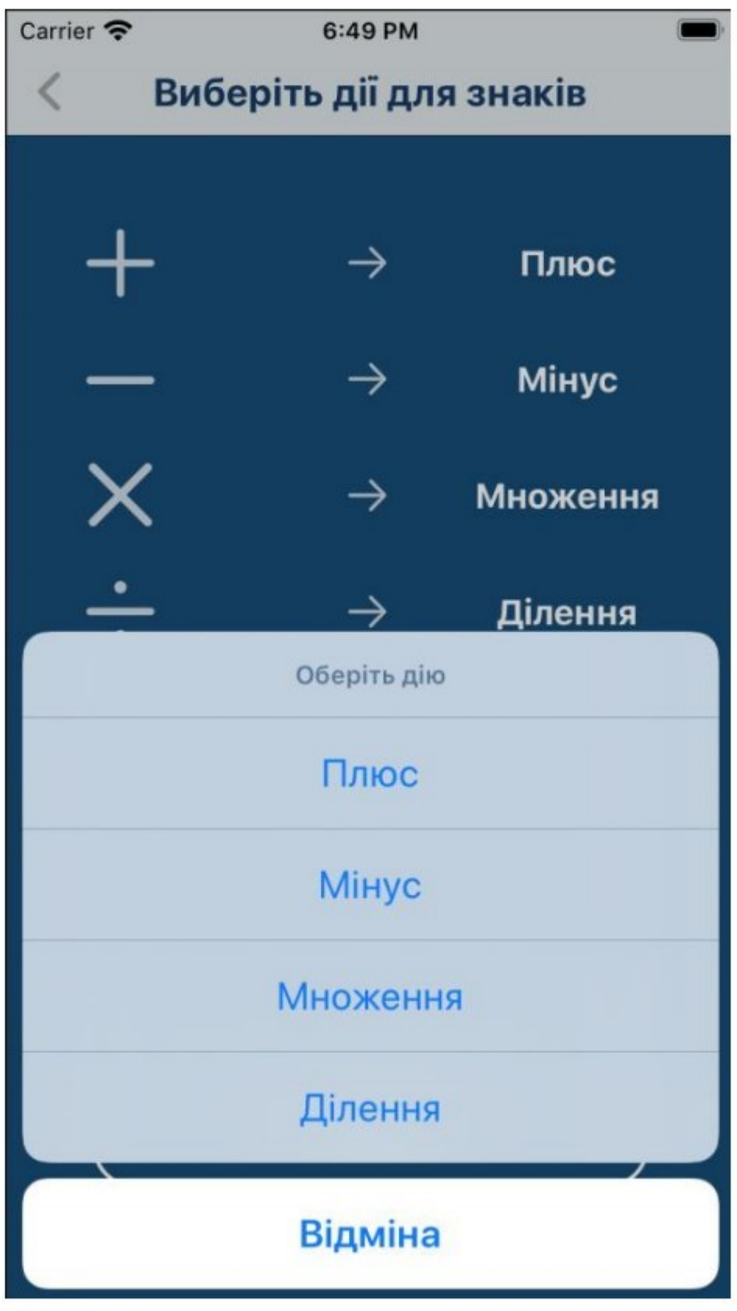


Рисунок 3.7 – Вигляд екрану налаштування знаку «+»

Контролер налаштування методу одноразових паролів містить кнопки збільшення/зменшення кількості символів в паролі, та кнопку «Далі». Максимальна кількість символів в паролі – 8, а мінімальна – 4. Вигляд вікна налаштування методу одноразових паролів показано на рисунку 3.8.

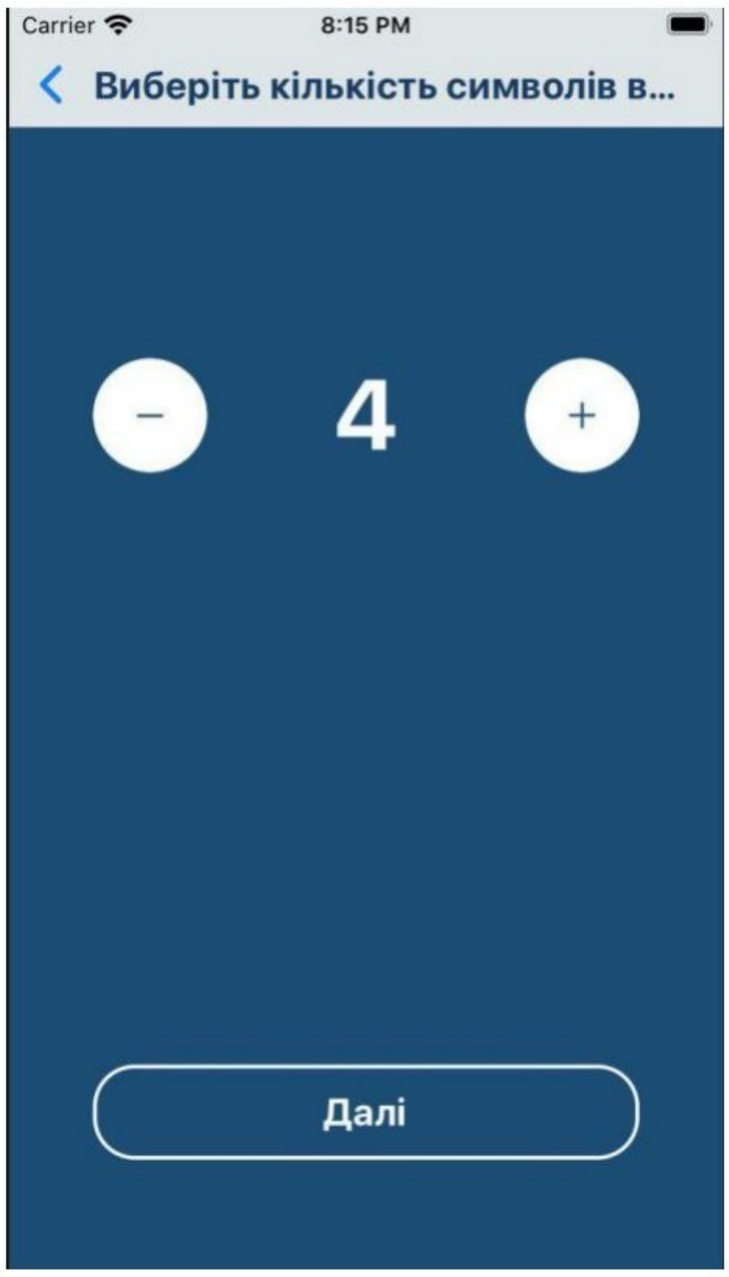


Рисунок 3.8 – Вигляд контролера налаштування методу одноразових паролів

3.3 Програмна реалізація та тестування блоку автентифікації

Після завершення процесу реєстрації та вибору і налаштування бажаних методів автентифікації, при наступному вході в застосунок користувачу слід виконати вхід за вибраним методом.

Екран входу за арифметичним методом містить сам приклад, який користувач повинен розв’язати та поле для введення відповіді. Для створення текстового поля з прикладом використано елемент UILabel():

```
let textLabel = UILabel()
textLabel.textColor = ColorScheme.kAppTextColor
textLabel.text = title
textLabel.font = .systemFont(ofSize: 20, weight: .bold)
```

А для поля для введення використано елемент UITextField():

```
var descTextView = UITextField()
descTextView.font = FontScheme.getSharjahFont(style: .bold, size: 16)
descTextView.textColor = UIColor.lightGray
descTextView.contentInset = UIEdgeInsets(top: textViewTopBottomInsert, left: editViewPadding, bottom: textViewTopBottomInsert, right: editViewPadding)
descTextView.delegate = self
```

Вигляд контролера з реалізацією арифметичного входу показано на рисунку 3.9.

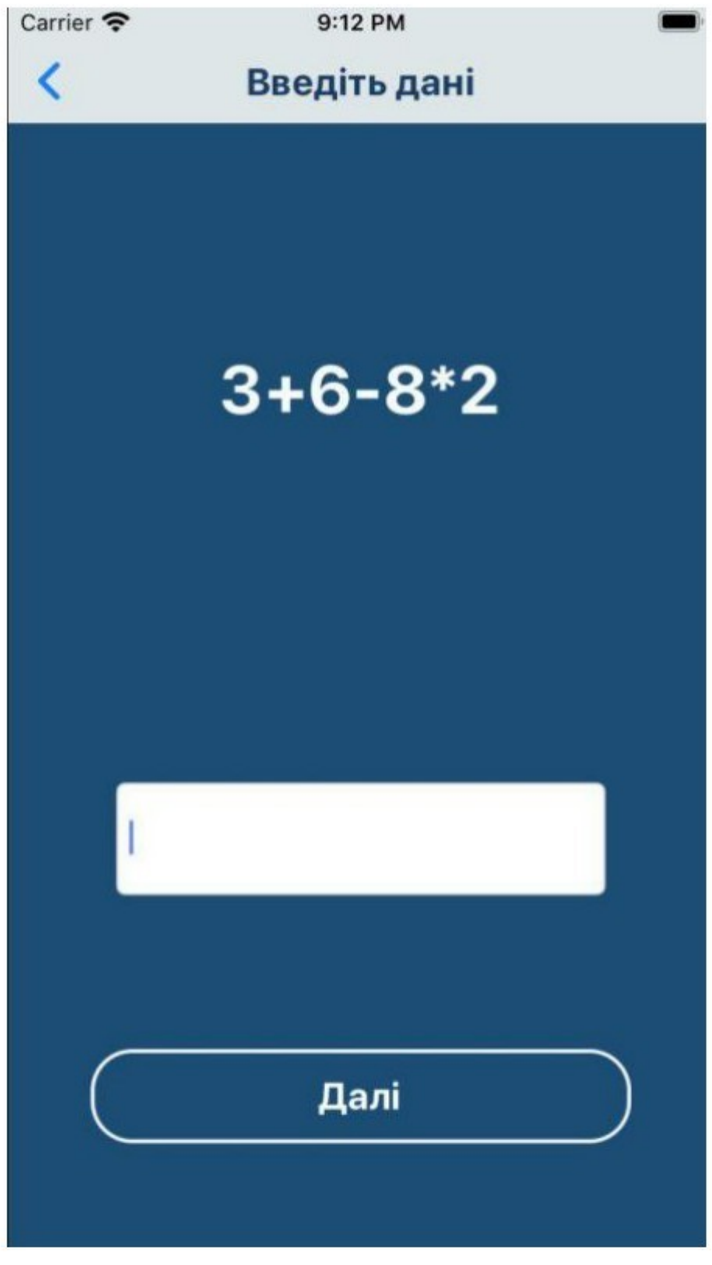


Рисунок 3.9 – Вигляд контролера з реалізацією арифметичного входу

Якщо введена користувачем відповідь не співпадає з правильною користувач отримує відповідне повідомлення (рис. 3.10).

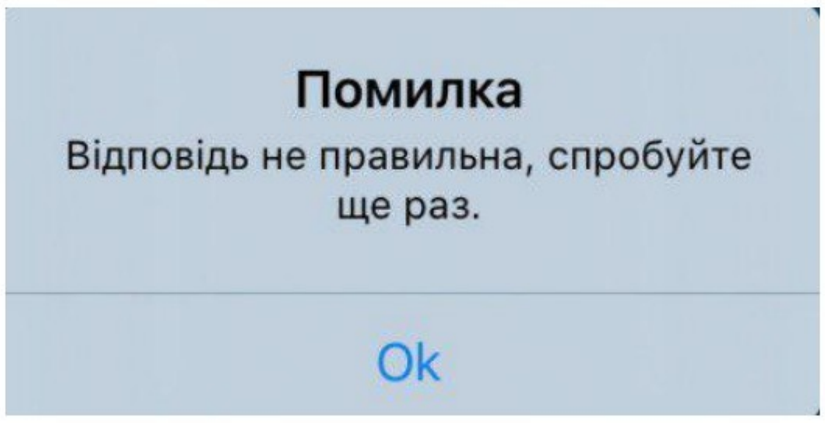


Рисунок 3.10 – Вигляд вікна повідомлення про неправильну відповідь

Екран входу за методом одноразових паролів містить поле для введення паролю, кнопку «Далі», а також кнопку «Відправити ще раз», яку слід натиснути якщо користувач не отримав пароль на пошту або якщо не правильно його ввів.

На рисунку 3.11 показано вигляд даного контролера.

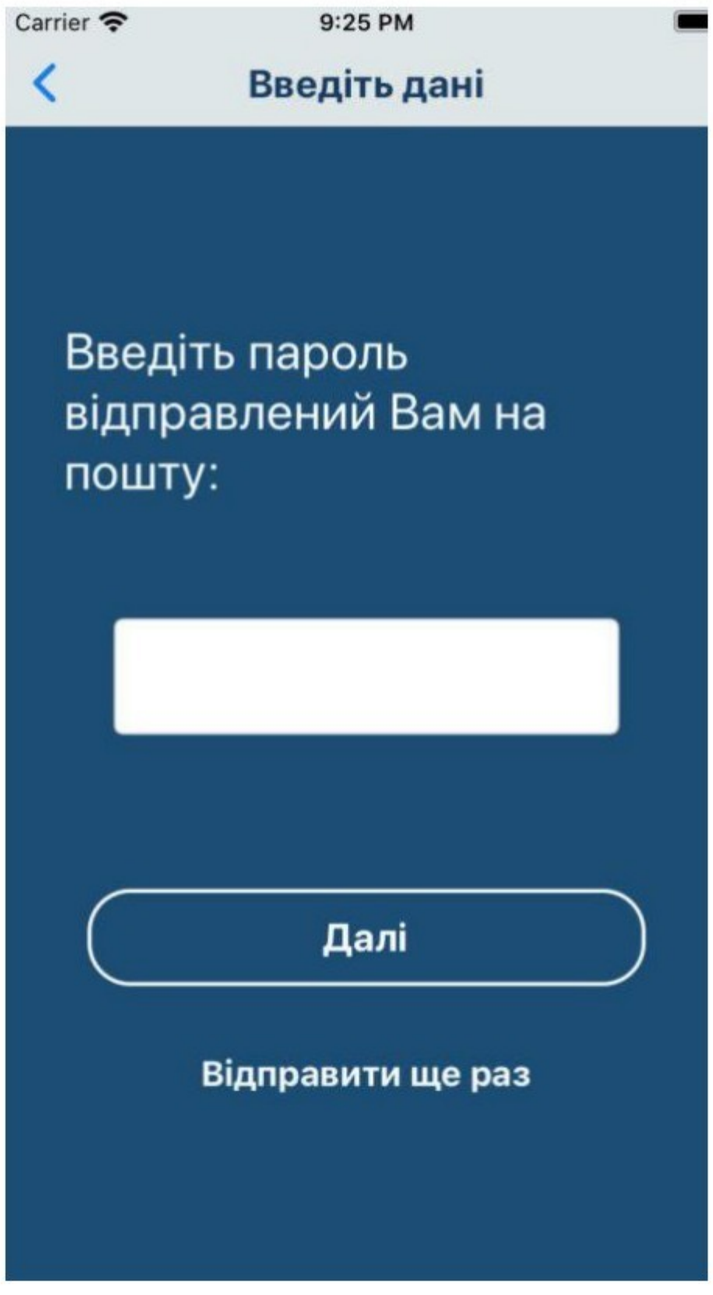


Рисунок 3.11 – Вигляд екрану з реалізацією входу за методом одноразових паролів

Якщо ж користувач вибрав комбінований метод авторизації, то йому слід пройти декілька етапів. Їх кількість залежить від налаштувань заданих при реєстрації. Серед можливих методів можуть бути: арифметичний (рис. 3.9), метод одноразових паролів (рис. 3.11), а також пароль або біометричний метод.

Екран з біометричною автентифікацію показано на рисунку 3.12.

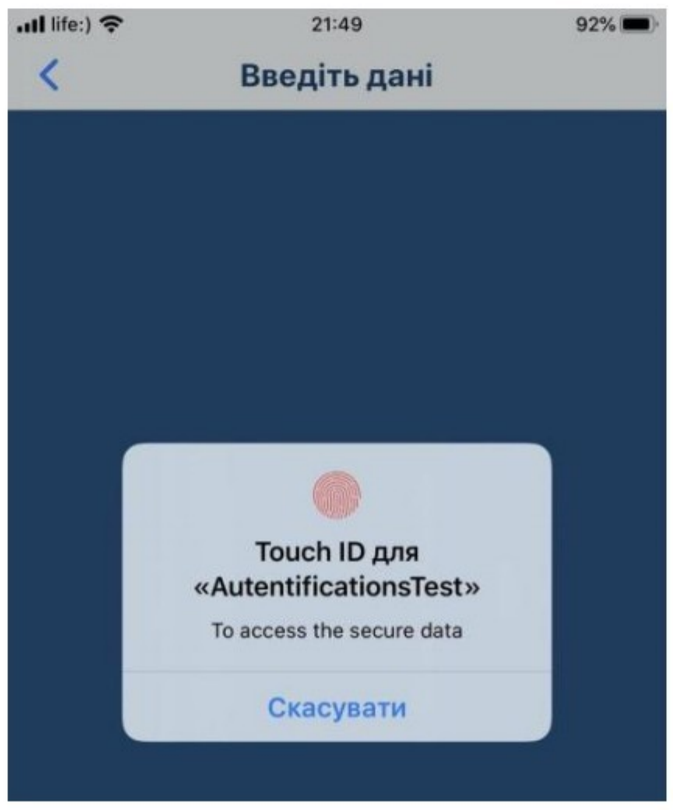


Рисунок 3.12 – Вигляд екрану з біометричною автентифікацією

Отже, за результатами експериментального дослідження, можна зробити висновок, що запропоновані методи автентифікації користувачів задовольняють поставлені до них вимоги, а також дозволяють покращити захист даних користувачів мобільних пристроїв, враховуючи вразливості наразі відомих методів.

Таке покращення було впроваджено і реалізовано у вигляді застосунку. Також дані методи можуть бути впровадженні у систему автентифікації самих пристроїв. Дані методи було реалізовано на основі аналізу головних недоліків відомих методів. Оскільки одним з головних недоліків усіх відомих методів є можливість підглядання під час введення то акцент було зроблено саме на цей недолік. Тому всі запропоновані методи вирішують дану проблему.

Наступним кроком після експериментального дослідження є економічне обґрунтування доцільності розробки.

4 ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ

4.1 Аналіз комерційного потенціалу розробки

4.1.1 Визначення рівня комерційного потенціалу розробки адаптивного методу для автентифікації користувачів мобільних пристроїв

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки адаптивного методу для автентифікації користувачів мобільних пристроїв. В результаті оцінювання можна буде зробити висновок щодо напрямів (особливостей) організації подальшого її впровадження з врахуванням встановленого рейтингу.

Для проведення технологічного аудиту залучимо 3-х незалежних експертів. У нашому випадку такими експертами будуть керівник магістерської роботи та провідні викладачі випускової та споріднених кафедр.

Оцінювання комерційного потенціалу розробки адаптивного методу для автентифікації користувачів мобільних пристроїв будемо здійснювати за 12-ю критеріями згідно рекомендацій.

Результати оцінювання комерційного потенціалу розробки адаптивного методу для автентифікації користувачів мобільних пристроїв заносимо до таблиці 4.1.

Таблиця 4.1. – Результати оцінювання комерційного успіху розробки адаптивного методу для автентифікації користувачів мобільних пристроїв

Критерії	Експерти		
	д. т. н., проф., Лужецький В.А.	к.т.н., ст. викл Лукічов В. В.	к.т.н., доцент Войтович О.П.
	Бали, виставлені експертами		
1	2	2	3
2	3	2	2
3	2	3	3
4	3	2	3

5	2	2	3
6	2	3	3
7	3	3	2
8	4	3	2
9	3	2	2
10	2	3	2
11	2	3	3
12	3	2	2
Сума балів	31	30	30
Середньоарифметична сума балів, СБ	30		

За даними таблиці 4.1 робимо висновок щодо рівня комерційного потенціалу розробки адаптивного методу для автентифікації користувачів мобільних пристроїв. При цьому користуємося рекомендаціями, наведеними в таблиці 4.2.

Таблиця 4.2 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0 – 10	Низький
11 – 20	Нижче середнього
21 – 30	Середній
31 – 40	Вище середнього
41 – 50	Високий

Таким чином, робимо висновок, щодо рівня комерційного потенціалу нашої розробки адаптивного методу для автентифікації користувачів мобільних пристроїв – середній.

4.1.2 Визначення рівня якості розробки адаптивного методу для автентифікації користувачів мобільних пристроїв

Оцінювання рівня якості розробки адаптивного методу для автентифікації користувачів мобільних пристрої впроводиться з метою порівняльного аналізу і визначення найбільш ефективного, з технічної точки зору, варіанта інженерного рішення.

Рівень якості – це кількісна характеристика міри придатності певного виду продукції для задоволення конкретного попиту на неї при порівнянні з відповідними базовими показниками за фіксованих умов споживання.

Абсолютний рівень якості розробки адаптивного методу для автентифікації користувачів мобільних пристроїв знаходимо обчисленням вибраних для її вимірювання показників, не порівнюючи їх із відповідними показниками аналогічних виробів.

Для цього необхідно визначити зміст основних функцій, які повинні реалізовувати розробка, вимоги замовника до неї, а також умови, які характеризують експлуатацію, визначають основні параметри, які будуть використані для розрахунку коефіцієнта технічного рівня виробу.

Система параметрів, прийнята до розрахунків, повинна достатньо повно характеризувати споживчі властивості інноваційного товару (його призначення, надійність, економічне використання ресурсів, стандартизація тощо). Далі визначаємо величину параметрів якості в балах та встановлюємо граничні його значення (кращі, гірші, середні).

Із врахуванням коефіцієнтів вагомості відповідних параметрів можна визначити абсолютний рівень якості інноваційного рішення за формулою:

$$К_{я.а.} = \sum_{i=1}^{n} Рні \cdot аі, (4.1)$$

де Рні – числове значення і-го параметра інноваційного рішення, n – кількість параметрів інноваційного рішення, що прийняті для оцінювання, аі –

коефіцієнт вагомості відповідного параметра (сума коефіцієнтів вагомості всіх параметрів повинна дорівнювати 1).

Всі ці дані для кожного параметра заносимо в табл. 4.3.

Таблиця 4.3 – Основні параметри адаптивного методу для автентифікації користувачів мобільних пристроїв

Параметри	Абсолютне значення параметра			Коефіцієнт вагомості параметра
	Краще +5…+4	Середнє +3	Гірше +1…+2	
Надійність методів			2	0,5
Доступність			2	0,3
Ресурсозатратність		3		0,1
Відноснапохибка		3		0,1

Отже, абсолютний рівень якості методу та засобу завадостійкого розподілу секрету становитиме – 2,2 бали.

Одночасно визначаємо відносний рівень якості адаптивного методу для автентифікації користувачів мобільних пристроїв, що виробляється (проектується), порівнюючи її показники з абсолютними показниками якості найліпших вітчизняних та зарубіжних аналогів (товарів-конкурентів) (табл. 4.4).

Відносний рівень якості методу та засобу завадостійкого розподілу секретувизначаємо за формулою:

$$К_{я.в.} = \sum_{i=1}^{n} qi \cdot ai, (4.2)$$

Таблиця 4.4 – Основні параметри адаптивного методу для автентифікації користувачів мобільних пристроїв та товару-конкурента

Параметри	Варіанти		Відносний показник якості	Коефіцієнт вагомості параметра
	Базовий (конкурент)	Новий		
Надійність методів	6	6	1	0,5
Доступність	2	4	2	0,3
Ресурсозатратність	6	5	0,83	0,1
Відноснапохибка	3	3	1	0,1

За розрахунками відносний рівень якості адаптивного методу для автентифікації користувачів мобільних пристроїв становитиме – 1,28. Це означає, що наша розробка краща за якістю на 28% від товару-аналога.

4.1.3 Визначення конкурентоспроможності розробки адаптивного методу для автентифікації користувачів мобільних пристроїв

У найширшому розумінні конкурентоспроможність товару – це можливість його успішного продажу на певному ринку і в певний проміжок часу.

Водночас конкурентоспроможною можна вважати лише однорідну продукцію з технічними параметрами і техніко-економічними показниками, що ідентичні аналогічним показникам уже проданого товару.

Для того, щоб високоякісний товар був одночасно і конкурентоспроможним, він має відповідати критеріям оцінювання споживачів конкретного ринку в конкретний час.

Дані для розрахунку загального показника конкурентоспроможності розробки необхідно занести до таблиці 4.5.

Таблиця 4.5 – Нормативні, технічні та економічні параметри адаптивного методу для автентифікації користувачів мобільних пристроїв і товару-конкурента

Параметри	Варіанти		Відносний показник якості	Коефіцієнт вагомості параметра
	Базовий (конкурент)	Новий		
Надійність методів	6	6	1	0,5
Доступність	2	4	2	0,3
Ресурсозатратність	6	5	0,83	0,1
Відноснапохибка	3	3	1	0,1
Ціна за продукт, тис. грн.	4500	2500	0,56	-

Загальний показник конкурентоспроможності розробки (К) з урахуванням вищезазначених груп показників визначаємо за формулою:

$$К = \frac{Іт.п.}{Іе.п.} = \frac{1,28}{0,56} = 2,9, \ (4.3)$$

де Іт.п. – індекс технічних параметрів (відносний рівень якості інноваційного рішення); Іе.п. – індекс економічних параметрів.

$$Іе.п. = \frac{РНеі}{РБеі} = \frac{2500}{4500} = 0,56, (4.4)$$

де РНеі, РБеі – економічні параметри (ціна придбання та споживання товару) відповідно нового та базового товарів.

Згідно розрахунків загальний показник конкурентоспроможності –2,9 . Це означає, що наша розробка адаптивного методу для автентифікації користувачів мобільних пристроїв більш конкурентна майже в3 рази від товару-аналога.

4.2 Прогнозування витрат на виконання науково-дослідної, дослідно-конструкторської та конструкторсько-технологічної роботи

4.2.1 Розрахунок витрат, що стосуються виконавців розробки адаптивного методу для автентифікації користувачів мобільних пристроїв

Основна заробітна плата кожного із розробників (дослідників) Зо, якщо вони працюють в наукових установах бюджетної сфери:

$$Зо = \frac{М}{Тр} \cdot t, (4.5)$$

де М – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.

У 2019 році величини окладів (разом з встановленими доплатами і надбавками) рекомендується брати в межах (5000…10000) грн. за місяць; Тр – число робочих днів в місяці; приблизно Тр = (21…23) дні; t – число робочих днів роботи розробника (дослідника).

Зроблені розрахунки зводимо до таблиці 4.6.

Таблиця 4.6 – Заробітна плата розробників

Посада	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник	10000	476	5	2380
Інженер-програміст	5000	238	5	1190
Всього:				3570

Основна заробітна плата робітників Зр, якщо вони беруть участь у виконанні даного етапу роботи і виконують роботи за робочими професіями у випадку, коли вони працюють в наукових установах бюджетної сфери, розраховується за формулою:

$$Зр = \sum_{i=1}^{n} ti \cdot Ci, \quad (4.6)$$

де ti – норма часу (трудомісткість) на виконання конкретної роботи, годин; n – число робіт по видах та розрядах; Сі – погодинна тарифна ставка робітника відповідного розряду, який виконує дану роботу. Сі визначається за формулою:

$$Ci = \frac{Мм \cdot Ki}{Тр \cdot Тзм}, \quad (4.7)$$

де Мм – розмір мінімальної заробітної плати за місяць, грн.; в 2019 році мінімальна заробітна плата становить – 4173 грн., Кі – тарифний коефіцієнт робітника відповідного розряду, Тр – число робочих днів в місяці; приблизно Тр = 21…23 дні; Тзм – тривалість зміни, зазвичай Тзм = 8 годин.

Таблиця 4.7 – Заробітна плата робітників

Найменування робіт	Трудомісткість, н-год.	Розряд роботи	Погодинна тарифна ставка	Тариф. коеф.	Величина, грн.
Програмувальні	10	7	38	1,54	380
Всього					380

Додаткова заробітна плата Зд всіх розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$Зд = 0{,}1 \cdot (Зр + Зо) = 0{,}1 \cdot (3570 + 380) = 395 \text{ грн.} \quad (4.8)$$

Нарахування на заробітну плату Нзп розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою:

де Зо – основна заробітна плата розробників, грн.; Зр – основна заробітна плата робітників, грн.; Зд – додаткова заробітна плата всіх розробників та робітників, грн.; β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, % (приймаємо для 1-го класу професійності ризику 22%).

$$Нзп = 0{,}22 \cdot (Зр + Зо + Зд) = 0{,}22 \cdot (3570 + 380 + 395) =$$

$$= 956 \text{ грн.} \quad (4.9)$$

Амортизація обладнання, комп'ютерів та приміщень А, які використовувались під час (чи для) виконання даного етапу роботи.

Дані відрахування розраховують по кожному виду обладнання, приміщенням тощо.

У спрощеному вигляді амортизаційні відрахування А в цілому бути розраховані за формулою:

$$А = \frac{Ц \cdot На}{100} \cdot \frac{Т}{12},$$

де Ц – загальна балансова вартість всього обладнання, комп'ютерів, приміщень тощо, що використовувались для виконання даного етапу роботи, грн.; На – річна норма амортизаційних відрахувань. Для нашого випадку можна прийняти, що На = (10...25)%; Т – термін, використання обладнання, приміщень тощо, місяці.

Таблиця 4.8 – Амортизаційні відрахування

Найменування	Ціна, грн.	Норма амортизації, %	Термін використання, м.	Сума амортизації
Смартфон	10000	20	3	500
Інше обладнання	2000	10	1	17
Всього	517			

Витрати на силову електроенергію Ве, якщо ця стаття має суттєве значення для виконання даного етапу роботи, розраховуються за формулою:

$$Ве = В \cdot П \cdot Ф \cdot Кп, грн$$

В – вартість 1 кВт-год. електроенергії, в 2019 р. В ≈ 8,45 грн./кВт; П – установлена потужність обладнання, кВт; Ф – фактична кількість годин роботи обладнання, годин, Кп – коефіцієнт використання потужності; Кп< 1.

Потужність обладнання складає –0,05 кВт;

Кількість годин роботи складає – 700годин;

Коефіцієнт викор. потужності -0,9;

$$Ве= 8,45 \cdot 0,05 \cdot 700 \cdot 0,9 =266 грн.$$

Інші витрати Він охоплюють: витрати на управління організацією, оплата службових відряджень, витрати на утримання, ремонт та експлуатацію основних засобів, витрати на опалення, освітлення, водопостачання, охорону праці тощо.

Інші витрати Ів можна прийняти як (100…300)% від суми основної заробітної плати розробників та робітників, які були виконували дану роботу, тобто:

$$Ів = 1 \cdot (Зо + Зр) = 1 \cdot (3570 + 380) = 3950 грн. \quad (4.10)$$

Сума всіх попередніх статей витрат дає витрати на виконання даної частини (розділу, етапу) роботи – В.

$$В = 3570 + 395 + 3950 + 266 + 956 + 517 + 380 = 10034 грн.$$

4.2.2 Розрахунок собівартості розробки адаптивного методу для автентифікації користувачів мобільних пристроїв

Витрати на силову електроенергію Ве, якщо ця стаття має суттєве значення для виконання даного етапу роботи, розраховуються за формулою:

$$Ве = В \cdot П \cdot Ф \cdot Кп, грн$$

В – вартість 1 кВт-год. електроенергії, в 2019 р. В ≈ 8,45 грн./кВт; П – установлена потужність обладнання, кВт; Ф – фактична кількість годин роботи обладнання, годин, Кп – коефіцієнт використання потужності; Кп< 1.

Потужність обладнання складає – 0,05 кВт;

Кількість годин роботи складає – 700 годин;

Коефіцієнт викор. потужності -0,9;

Ве = 266 грн;

Основна заробітна плата робітників Зр, якщо вони беруть участь у виконанні даного етапу роботи і виконують роботи за робочими професіями у випадку, коли вони працюють в наукових установах бюджетної сфери, розраховується за формулою:

$$Зр = \sum_{i=1}^{n} ti \cdot Ci, (4.11)$$

де ti – норма часу (трудомісткість) на виконання конкретної роботи, годин; n – число робіт по видах та розрядах; Сі – погодинна тарифна ставка робітника відповідного розряду, який виконує дану роботу. Сі визначається за формулою:

$$Ci = \frac{Мм \cdot Кі}{Тр \cdot Тзм}, \quad (4.12)$$

де Мм – розмір мінімальної заробітної плати за місяць, грн.; в 2019 році мінімальна заробітна плата становить – 4173 грн., Кі – тарифний коефіцієнт робітника відповідного розряду, Тр – число робочих днів в місяці; приблизно Тр = 21…23 дні; Тзм – тривалість зміни, зазвичай Тзм = 8 годин

Таблиця 4.9 – Заробітна плата робітників

Найменування робіт	Трудомісткість, н-год.	Розряд роботи	Погодинна тарифна ставка	Тариф. коеф.	Величина, грн.
Програмувальні	10	7	38	1,54	380
Всього					380

Додаткова заробітна плата Зд всіх робітників, які брали участь у виконанні даного етапу роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$Зд = 0{,}1 \cdot (Зо) = 0{,}1 \cdot (380) = 38 \text{ грн.} \quad (4.13)$$

Нарахування на заробітну плату Нзп розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою:

де Зо – основна заробітна плата розробників, грн.; Зр – основна заробітна плата робітників, грн.; Зд – додаткова заробітна плата всіх розробників та робітників, грн.; β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, % (приймаємо для 1-го класу професійності ризику 22%).

$$Нзп = 0{,}22 \cdot (Зо + Зд) = 0{,}22 \cdot (380 + 38) = 92 \text{ грн.} \quad (4.14)$$

«Загальновиробничі витрати» належать витрати: пов'язані з управлінням виробництвом (утримання працівників апарату управління виробництвом, оплата службових відряджень персоналу цехів, витрати на інформаційне забезпечення управління тощо); на повне відновлення та капітальний ремонт основних фондів загальновиробничого призначення; витрати некапітального характеру, пов'язані з удосконаленням технологій та організацією виробництва, поліпшенням якості продукції; на утримання, обслуговування, поточний ремонт виробничих приміщень; на контроль за виробничими процесами та кістю продукції.

Крім того, загальновиробничі витрати з розрахунку на одиницю продукції можна розрахувати за нормативами відносно до основної заробітної плати основних робітників, які виготовляють продукцію:

$$ЗВВ = Нв \cdot Зо, \quad (4.15)$$

Норматив загальновиробничих витрат для програмних продуктів становить 230-270%.

$$ЗВВ = 2{,}5 \cdot 380 = 950 \text{ грн},$$

Сума попередніх витрат утворює виробничу собівартість розробки:

$$S_В = 1726 \text{ грн}.$$

4.3 Розрахунок ціни та чистого прибутку від реалізації розробки адаптивного методу для автентифікації користувачів мобільних пристроїв

Ціна – це грошовий вираз вартості товару (продукції, послуги). Вона завжди коливається навколо ціни виробництва (перетвореної форми вартості одиниці товару, що дорівнює сумі витрат виробництва й середнього прибутку) та відображає рівень суспільне необхідних витрат праці.

Виходячи з того, що розробки, як правило, приймаються та впроваджуються за завданням замовника, або коли результатом розробки є продукція, що підлягає державному регулюванню, то нижню межу ціни реалізації розробки можна розрахувати за формулою:

$$Ц = S_В \cdot \left(1 + \frac{Р}{100}\right) \cdot \left(1 + \frac{\omega}{100}\right), \quad (4.16)$$

де Sв – виробнича собівартість інноваційного рішення, грн.; Р – норматив рентабельності узгоджений із замовником або встановлений державою, (Р=30…60%); w – ставка податку на додану вартість, % (в 2019 році w=20%).

$$Ц = 1726 \cdot \left(1 + \frac{30}{100}\right) \cdot \left(1 + \frac{20}{100}\right) = 2692 \text{ грн}.$$

Чистий прибуток від реалізації розробки можна розрахувати за формулою:

$$П = \left(Ц - \frac{(Ц - МР) \cdot f}{100} - S_В - \frac{q \cdot S_В}{100}\right) \cdot \left(1 - \frac{h}{100}\right) \cdot РП, \quad (4.17)$$

де Ц – ціна розробки, грн.; МР – вартість матеріальних та інших ресурсів, що були придбані виробником для виготовлення розробки (МР=(0,1…0,2) Цр), грн.; f – зустрічна ставка податку на додану вартість, %; Sв – виробнича

собівартість розробки, грн.; q – норматив, який визначає величину адміністративних витрат, витрат на збут та інші операційні витрати, % (рекомендовано q=5…10%); h – ставка податку на прибуток, %, РП – прогнозований попит продажів:

$$П = 7728 \text{ грн.}$$

4.4 Розрахунок терміну окупності коштів, вкладених в наукову розробку адаптивного методу для автентифікації користувачів мобільних пристроїв

Термін окупності вкладених у реалізацію наукового проекту інвестицій Ток можна розрахувати за формулою:

$$Ток = \frac{В}{П} \quad (4.18)$$

Отже, результат становить:

$$Ток = \frac{10034}{7728} = 1{,}3 \text{ роки.}$$

Оскільки Ток < 3 років, то фінансування даної наукової розробки адаптивного методу для автентифікації користувачів мобільних пристроїв є доцільним.

ВИСНОВКИ

У магістерській кваліфікаційній роботі проведено аналіз вразливостей системи автентифікації і показано недоліки існуючих засобів автентифікації користувачів операційної системи iOS.

Аналіз інформаційних джерел показав, що існує три основних методи для автентифікації користувачів в ОС iOS. Кожний з методів має свої переваги та недоліки, різну стійкість до зламу. Найкращим наразі є біометричний метод, який найважче піддається зламу, хоча і він має ряд недоліків. Досліджено структуру безпеки операційної системи, а також наведено та описано архітектуру безпеки iOS.

На основі аналізу виявлено, що кожний з існуючих методів автентифікації має ряд недоліків, через які зловмисник відносно легко може отримати доступ до персональних даних користувачів.

Оскільки через мобільні пристрої користувач може отримати доступ до багатьох застосунків, серед яких фінансові, за допомогою яких можна проводити будь які банківські операції, доцільно підвищувати захищеність, як мобільних пристроїв, так і самих застосунків. Відповідно до цього була визначена мета магістерської кваліфікаційної роботи, що полягає у покращені методів захисту інформації та створенні програмного засобу для удосконалення системи безпеки операційної системи iOS.

Запропоновано та розроблено адаптивний метод для автентифікації користувачів мобільних пристроїв. На основі запропонованого методу були розроблені три методи автентифікації користувачів:

- арифметичний;
- метод одноразових паролів;
- омбінований метод.

Головною особливістю розроблених методів є те, що вони значно ускладнюють можливість підглядання під час введення, а також не потребуються ніякого додаткового обладнання для реалізації.

Результатом виконання магістерської кваліфікаційної роботи є програмний засіб, виконаний у середовищі xCode мовою Swift. Для розробки засобу підготовлений ряд схем і алгоритмів, розроблений інтерфейс, реалізовано цілу низку окремих функцій для виконання конкретних операцій. Розроблений програмний засіб протестований з метою перевірки ефективності здійснюваного захисту.

Розроблений програмний засіб може бути використаний для автентифікації користувачів в будь яких застосунках або в самій ОС iOS.

Наведено економічне обґрунтування, за яким розробка визнана доцільною для впровадження, оскільки за прогнозами вона принесе прибуток у розмірі 7728 грн. за 3 роки, а термін окупності становитиме 1,5 року, що є досить високим показником для інвестування.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Актуальные киберугрозы II квартал 2018 года. [Електронний ресурс]. https://www.ptsecurity.com/ru-ru/research/analytics/cybersec-threatscape-2018-q2/.

2. Рынок преступных кибероуслуг. [Електронний ресурс]. https://www.ptsecurity.com/ru-ru/research/analytics/darkweb-2018/ (дата звернення: 07.10.2018).

3. Лукічов В. В. Методи та засоби стеганографічного захисту інформації в комп'ютерних системах і мережах на основі вейвлет-перетворень : дис. канд. техн. наук / Лукічов Віталій Володимирович – Вінниця, 2010. – 204 с.

4. Лукічов В. В. Метод шаблонного вбудовування даних у вейвлет коефіцієнти на основі критерію стеганографічної стійкості / В. В. Лукічов, А. С. Васюра – Вінниця, 2010. – 8 с.

5. Безопасность iOS [Електронний ресурс] / – Режим доступу до ресурсу: https://www.securitylab.ru/contest/428454.php.

6. Использование паролей для програм [Електронний ресурс] / – Режим доступу до ресурсу: https://support.apple.com/ru-ru/HT204397.

7. iOS 12.4 File System Extraction [Електронний ресурс] / – Режим доступу до ресурсу: https://blog.elcomsoft.com/2019/09/ios-12-4-file-system-extraction/.

8. Безопасность iOS и как ее усилить [Електронний ресурс] / – Режим доступу до ресурсу: https://xakep.ru/2018/06/27/ios-security/.

9. Бирюков А. А. Информационная безопасность: защита и нападение / А. А. Бирюков., 2017. – 576 с – ISBN 978-5- 97060-435-9

10. Лужецький В. А. Інформаційна безпека : навчальний посібник / В. А. Лужецький, О. П. Войтович, А. В. Дудатьєв – Вінниця : УНІ-ВЕРСУМ-Вінниця, 2009. – 240 с. – ISBN 978-966-641-265-5

11. Безопасность iOS: защита файлов [Електронний ресурс] / – Режим доступу до ресурсу: https://touchin.ru/letters/bezopasnost-ios-zashchita-failov/.

12. AES Advanced Encryption Standard [Електронний ресурс] / – Режим доступу до ресурсу: https://ua.nettech.ua/news/aes-advanced-encryption-standard-ili-rijndael.

13. Руководство по языку программирования Swift [Електронний ресурс] / – Режим доступу до ресурсу: https://metanit.com/swift/tutorial/.

14. My Development Toolset 2019 for iOS [Електронний ресурс] / - – Режим доступу до ресурсу: https://medium.com/duruldalkanat/my-development-toolset-2017-for-ios-7c0758e3e5ce.

15. Навигация Средствами Контроллера [Електронний ресурс] / – Режим доступу до ресурсу: https://swiftbook.ru/post/tutorials/ios-root-controller-navigation/.

Додаток А

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра захисту інформації

ЗАТВЕРДЖУЮ

Завідувач кафедри ЗІ, д. т. н., проф.

_____ В. А. Лужецький

"____" _____ 2019 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до магістерської кваліфікаційної роботи на тему:

"Адаптивний метод автентифікації користувачів мобільних пристроїв"

08-20.МКР.002.00.000 ТЗ

Керівник магістерської кваліфікаційної

роботи к. т. н., ст. викл.

_____ В. В. Лукічов

«___»_____2019 р.

Вінниця 2019

1 Найменування та область застосування

Метод і засіб для захисту від несанкціонованого доступу за допомогою адаптивного метода автентифікації користувачів. Область використання: пристрої з ОС iOS.

2 Підстави для розробки

Розробка виконується на основі наказу № 254 ректора ВНТУ від 02.10.2019 р..

3 Мета та призначення

Підвищення ефективності захисту від несанкціонованого доступу шляхом удосконалення системи автентифікації ОС iOS.

4 Джерела розробки

– Федорченко В. Н. Анализ угроз для мобильных устройств и способов их защиты / В. Н. Федорченко,Н. Ю. Шевякова – М.: Бином, 2013. – 68 с.

– Защита данных на телефонах и планшетах на базе ОС iOS [Електронний ресурс]. – Режим доступу : URL : https://rohos.com/2013/ios-security/ – Назва з екрану.

– Основы безопасности операционной системы iOS. Уровень ядра [Електронний ресурс]. – Режим доступу: URL: http://habrahabr/post/I 75517/. – Назва з екрану.

– Ярочкин В. А. Информационная безопасность / Владимир Ярочкин. – М: Гаудемаус, 2-е изд. 2004, – 544 с.

5 Технічні вимоги

5.1 Параметри розроблюваної системи захисту:

– механізм захисту – автономний додаток;
– принцип захисту – захист від несанкціонованого доступу;
– метод захисту – автентифікація користувача за адаптивним методом;
– об’єкт захисту – дані мобільного пристрою на базі ОС iOS.

5.2 Програма повинна працювати без помилок і давати змогу:

– програма повинна автентифікувати користувачів;
– отримувати доступ до інформації при виконанні відповідних вимог;

5.3 Програма має бути простою у використанні, для швидкого та простого введення змін та оснащеною системою допомоги.

5.4 Вимоги до апаратного і програмного забезпечення, на якому повинна працювати програма:

– оперативна пам’ять – не менше 512 Мбайт;

– обсяг зовнішньої пам’яті для розташування додаткових файлів – не менше 4,81 Mb;

– середовище функціонування – операційна система – iOS 9.0 і вище.

5.5 Вимоги до техніки безпеки при роботі з програмою повинні відповідати існуючим вимогам та стандартам з техніки безпеки при користуванні комп’ютерною технікою.

6 Стадії та етапи розробки

№	Зміст	Початок	Закінчення	Результат
1	Вступ. Розробка ТЗ. Огляд літературних джерел	01.09.2019	22.09.2019	Розділ пояснювальної записки, технічне завдання
2	Розробка схеми роботи програми в цілому і алгоритмів її складових. Вибір програмних засобів для реалізації завдання.	23.09.2019	12.10.2019	Розділ пояснювальної записки
3	Програмна реалізація	14.10.2019	17.11.2019	Модулі програмного засобу
	Тестування засобу	18.11.2019	24.11.2019	Діюча програма. Пояснювальна записка
4	Аналіз виконання ТЗ, висновки. Оформлення пояснювальної записки	25.11.2019	30.11.2019	Пояснювальна записка

7 Порядок контролю та приймання

7.1 До приймання дипломної роботи представляється:

– ПЗ до магістерської кваліфікаційної роботи;

– програмний засіб розподілу секрету;

– результати тестування;

– ілюстративні матеріали для захисту.

7.2 Рубіжний контроль керівника _____

7.3 Попередній захист на кафедрі _____

7.4 Захист на ДЕК _____

Додаток Б

Лістинг програмного засобу

Файл ChooseAuthVC.swift

```
import UIKit

protocol ChooseAuthViewControllerProtocol: class {

}

class ChooseAuthViewController: UIViewController {

// MARK: - IBOutlets
@IBOutlet weak var tableView: UITableView!

// MARK: - Properties

var presenter: ChooseAuthPresenterProtocol!


// MARK: - Lifecycle

override func viewDidLoad() {
    super.viewDidLoad()

    presenter = ChooseAuthPresenter(view: self)

    setupViews()
}

// MARK: - UI

private func setupViews() {
    setupNavBar()
    setupTableView()
}

func setupTableView() {
    tableView.delegate = self
    tableView.dataSource = self

    tableView.registerCell(AuthTypeTableViewCell.self)
}

func setupNavBar() {
    let titleLabel = UILabel()

    let title = "Виберіть метод захисту"

    titleLabel.textColor = ColorScheme.kAppTextColor
        titleLabel.text = title
        titleLabel.font = .systemFont(ofSize: 20, weight: .bold)
        navigationItem.titleView = titleLabel

        navigationItem.leftBarButtonItem = UIBarButtonItem(image: UIImage(named: "back"), style: .plain,
target: self, action: #selector(self.backTapped))
        navigationController?.navigationBar.barTintColor = ColorScheme.kBackgroundColor
    }

    // MARK: - IBActions

    @objc func backTapped() {
        self.navigationController?.popViewController(animated: true)
    }
```

```
}

extension ChooseAuthViewController: UITableViewDataSource {

    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        return 3
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
        let cell = tableView.createCell(AuthTypeTableViewCell.self, indexPath)
        presenter.configurateCell(cell: cell, row: indexPath.row)
        return cell
    }

    func tableView(_ tableView: UITableView, viewForHeaderInSection section: Int) -> UIView? {
        let view = UIView()
        view.backgroundColor = .clear
        return view
    }

    func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
        print("\(indexPath.row) row selected")

        let vc = OncePasswordViewController.instance(.main)
        navigationController?.pushViewController(vc, animated: true)
    }

}

extension ChooseAuthViewController: UITableViewDelegate {

    func tableView(_ tableView: UITableView, heightForRowAt indexPath: IndexPath) -> CGFloat {
        return 100
    }

    func tableView(_ tableView: UITableView, heightForHeaderInSection section: Int) -> CGFloat {
        return CGFloat.leastNonzeroMagnitude
    }
}

extension ChooseAuthViewController: ChooseAuthViewControllerProtocol {

}
```

Файл BiometricVC.swift

```
import UIKit
import LocalAuthentication

protocol BiometricLoginViewControllerProtocol: class {

}

class BiometricLoginViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
        authenticationWithTouchID()
        setupNavBar()
    }

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)

        self.navigationController?.isNavigationBarHidden = false
    }
```

```
override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning()
    // Dispose of any resources that can be recreated.
}

func setupNavBar() {
    let titleLabel = UILabel()

    let title = "Введіть дані"

    titleLabel.textColor = ColorScheme.kAppTextColor
    titleLabel.text = title
    titleLabel.font = .systemFont(ofSize: 20, weight: .bold)
        navigationItem.titleView = titleLabel

        navigationItem.leftBarButtonItem = UIBarButtonItem(image: UIImage(named: "back"), style: .plain,
target: self, action: #selector(self.backTapped))
        navigationController?.navigationBar.barTintColor = ColorScheme.kBackgroundColor
    }

    @objc func backTapped() {
        self.navigationController?.popViewController(animated: true)
    }
}

extension BiometricLoginViewController {

    func authenticationWithTouchID() {
        let localAuthenticationContext = LAContext()
        localAuthenticationContext.localizedFallbackTitle = "Use Passcode"

        var authError: NSError?
        let reasonString = "To access the secure data"

        if localAuthenticationContext.canEvaluatePolicy(.deviceOwnerAuthenticationWithBiometrics, error:
&authError) {

            localAuthenticationContext.evaluatePolicy(.deviceOwnerAuthenticationWithBiometrics,
localizedReason: reasonString) { success, evaluateError in

                if success {

                    //TODO: User authenticated successfully, take appropriate action

                } else {
                    //TODO: User did not authenticate successfully, look at error and take appropriate action
                    guard let error = evaluateError else {
                        return
                    }

                    print(self.evaluateAuthenticationPolicyMessageForLA(errorCode: error._code))

                    //TODO: If you have choosen the 'Fallback authentication mechanism selected'
(LAError.userFallback). Handle gracefully

                }
            }
        } else {

            guard let error = authError else {
                return
            }
            //TODO: Show appropriate alert if biometry/TouchID/FaceID is lockout or not enrolled
            print(self.evaluateAuthenticationPolicyMessageForLA(errorCode: error.code))
        }
    }
```

```
    func evaluatePolicyFailErrorMessageForLA(errorCode: Int) -> String {
        var message = ""
        if #available(iOS 11.0, macOS 10.13, *) {
            switch errorCode {
                case LAError.biometryNotAvailable.rawValue:
                    message = "Authentication could not start because the device does not support biometric
authentication."

                case LAError.biometryLockout.rawValue:
                    message = "Authentication could not continue because the user has been locked out of
biometric authentication, due to failing authentication too many times."

                case LAError.biometryNotEnrolled.rawValue:
                    message = "Authentication could not start because the user has not enrolled in biometric
authentication."

                default:
                    message = "Did not find error code on LAError object"
            }
        } else {
            switch errorCode {
case LAError.touchIDLockout.rawValue:
                    message = "Too many failed attempts."

                case LAError.touchIDNotAvailable.rawValue:
                    message = "TouchID is not available on the device"

                case LAError.touchIDNotEnrolled.rawValue:
                    message = "TouchID is not enrolled on the device"

                default:
                    message = "Did not find error code on LAError object"
            }
        }

        return message;
    }

    func evaluateAuthenticationPolicyMessageForLA(errorCode: Int) -> String {

        var message = ""

        switch errorCode {

        case LAError.authenticationFailed.rawValue:
            message = "The user failed to provide valid credentials"

        case LAError.appCancel.rawValue:
            message = "Authentication was cancelled by application"

        case LAError.invalidContext.rawValue:
            message = "The context is invalid"

        case LAError.notInteractive.rawValue:
            message = "Not interactive"

        case LAError.passcodeNotSet.rawValue:
            message = "Passcode is not set on the device"

        case LAError.systemCancel.rawValue:
            message = "Authentication was cancelled by the system"

        case LAError.userCancel.rawValue:
            message = "The user did cancel"
```

```
        case LAError.userFallback.rawValue:
            message = "The user chose to use the fallback"

        default:
            message = evaluatePolicyFailErrorMessageForLA(errorCode: errorCode)
        }

        return message
    }
}

extension BiometricLoginViewController: BiometricLoginViewControllerProtocol {

}
```

Файл NumberVC.swift

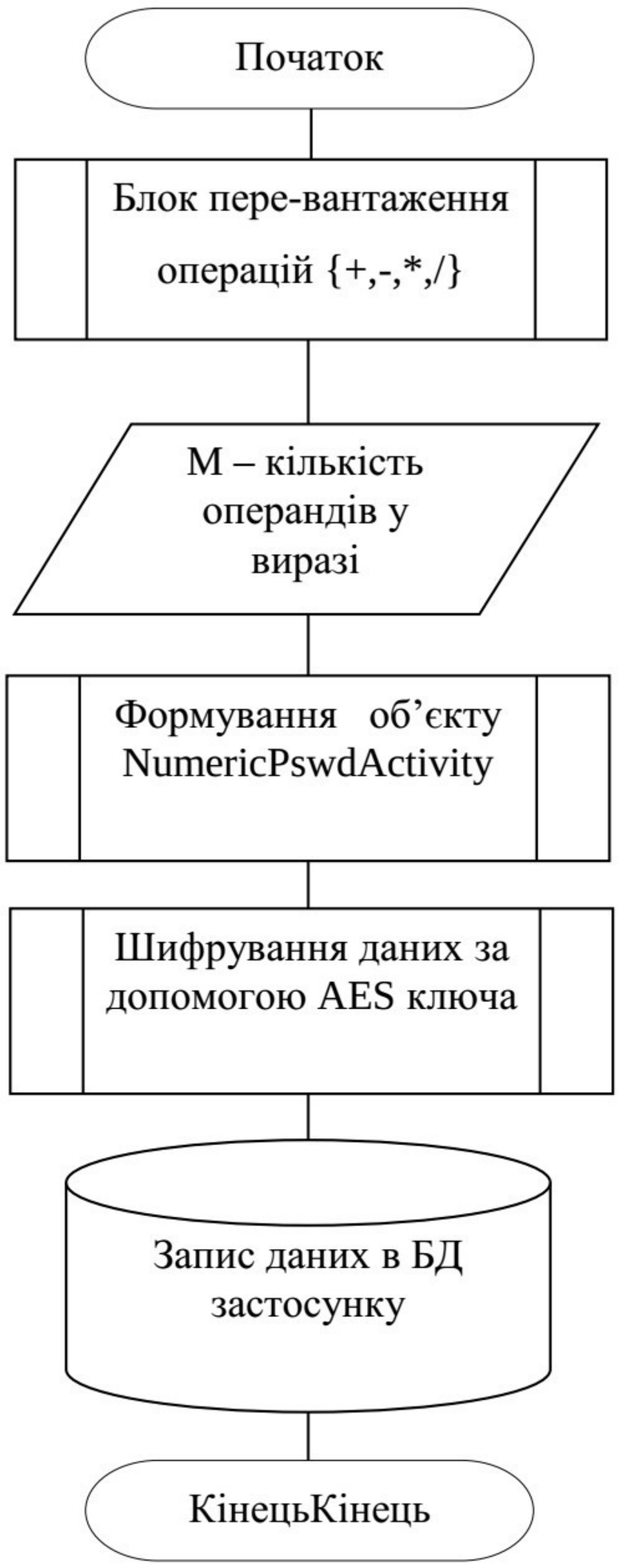
```
import UIKit

protocol NumberViewControllerProtocol: class {

}

class NumberViewController: UIViewController {

    // MARK: - IBOutlets

    @IBOutlet weak var plusButton: UIButton!

    @IBOutlet weak var minusButton: UIButton!

    @IBOutlet weak var multyplyButton: UIButton!

    @IBOutlet weak var divButton: UIButton!

    @IBOutlet weak var nextButton: UIButton!


    // MARK: - Properties


    var presenter: NumberPresenterProtocol!


    private let cornerRadius: CGFloat = 22

    private let borderWidth: CGFloat = 2


    // MARK: - Lifecycle


     override func viewDidLoad() {

         super.viewDidLoad()

         presenter = NumberPresenter(view: self)

         setupViews()

     }


    private func setupViews() {
```

```
        setupNavBar()

        setupButtons()

    }


    func setupNavBar() {

        let titleLabel = UILabel()

        let title = "Виберіть дії для знаків"

        titleLabel.textColor = ColorScheme.kAppTextColor

        titleLabel.text = title

        titleLabel.font = .systemFont(ofSize: 20, weight: .bold)

        navigationItem.titleView = titleLabel



        navigationItem.leftBarButtonItem = UIBarButtonItem(image: UIImage(named: "back"), style: .plain,
    target: self, action: #selector(self.backTapped))

        navigationController?.navigationBar.barTintColor = ColorScheme.kBackgroundColor

    }

    func setupButtons() {

        nextButton.viewBorder(color: .white, width: borderWidth)

        nextButton.viewCorner(cornerRadius)


        plusButton.setTitle("Плюс", for: .normal)

        minusButton.setTitle("Мінус", for: .normal)

        multyplyButton.setTitle("Множення", for: .normal)

        divButton.setTitle("Ділення", for: .normal)

    }

    @objc func backTapped() {

        self.navigationController?.popViewController(animated: true)

    }

    func displayActionSheet(button: UIButton) {

         // 1

         let optionMenu = UIAlertController(title: nil, message: "Оберіть дію", preferredStyle: .actionSheet)

        // 2

         let plusAction = UIAlertAction(title: "Плюс", style: .default, handler: {

             (alert: UIAlertAction!) in

            button.setTitle("Плюс", for: .normal)

         })

         let minusAction = UIAlertAction(title: "Мінус", style: .default, handler: {

             (alert: UIAlertAction!) in
```

```
    button.setTitle("Мінус", for: .normal)
})

let multyAction = UIAlertAction(title: "Множення", style: .default, handler: {
    (alert: UIAlertAction!) in
    button.setTitle("Множення", for: .normal)
})

let divAction = UIAlertAction(title: "Ділення", style: .default, handler: {
    (alert: UIAlertAction!) in
    button.setTitle("Ділення", for: .normal)
})
// 3
let cancelAction = UIAlertAction(title: "Відміна", style: .cancel, handler: {
(alert: UIAlertAction!) in
})
// 4
optionMenu.addAction(plusAction)
     optionMenu.addAction(minusAction)
     optionMenu.addAction(multyAction)
     optionMenu.addAction(divAction)
     optionMenu.addAction(cancelAction)
     // 5\
     self.present(optionMenu, animated: true, completion: nil)
 }

// MARK: - IBActions
@IBAction func plusButtonClick(_ sender: UIButton) {
    displayActionSheet(button: sender)
}
@IBAction func minusButtonClick(_ sender: UIButton) {
    displayActionSheet(button: sender)
}
@IBAction func multyButtonClick(_ sender: UIButton) {
    displayActionSheet(button: sender)
}
}
```

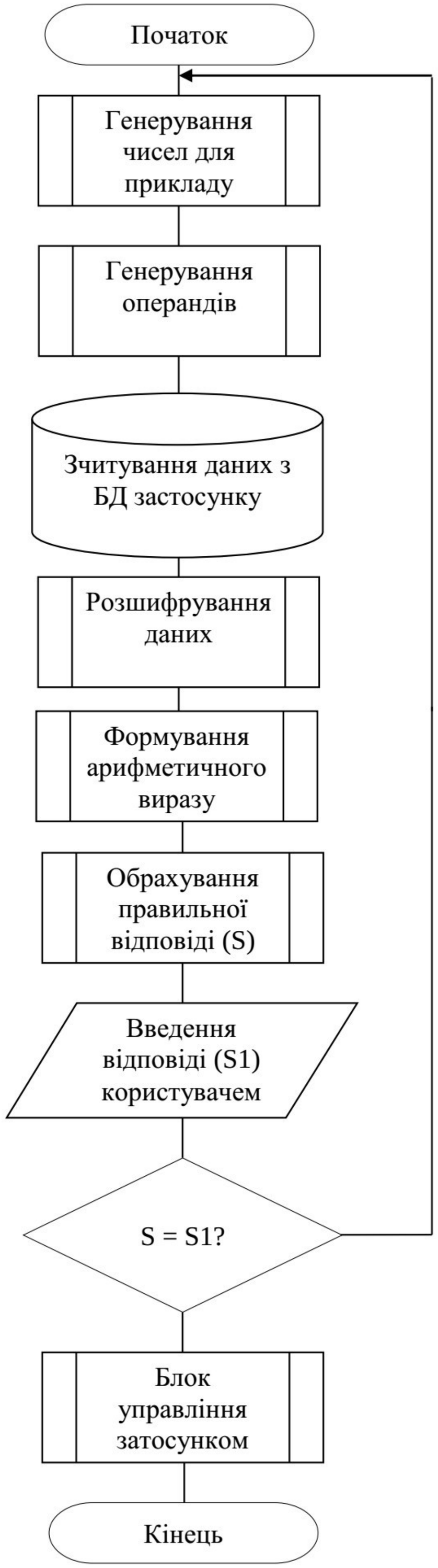
ІЛЮСТРАТИВНА ЧАСТИНА

Схема налаштування арифметичного методу



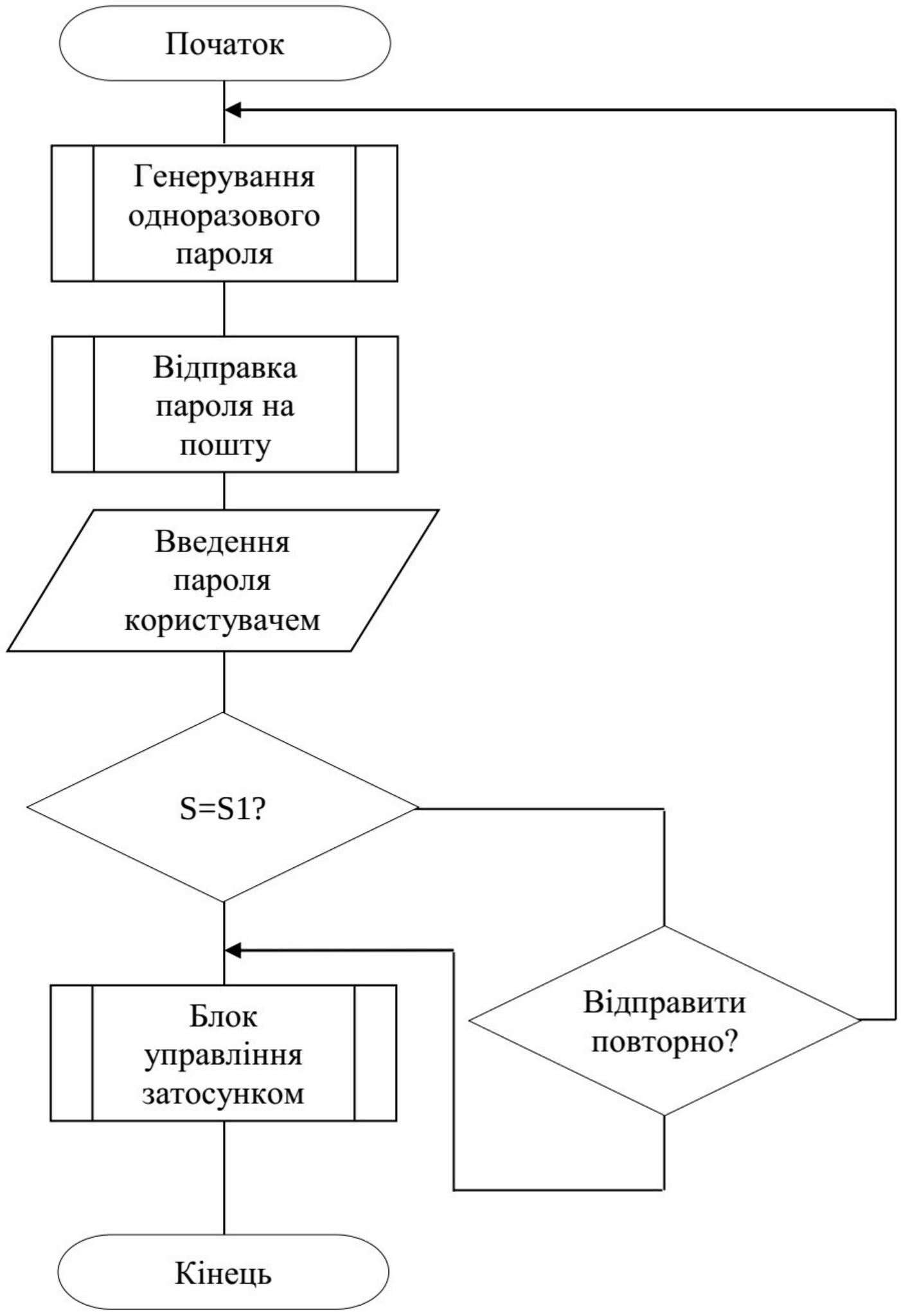
					08-20.МКР.002.00.000 141			
Зм	Арк.	№ докум.	Підпис	Дат				
Розроб.		Вітович М.М			Адаптивний метод автентифікації користувачів мобільних	Літ.	Маса	Масштаб
Перевір.		Лукічов В.В						
Реценз.		Крупельниць				ВНТУ, гр. 1 БС-		
Н. Контр.		Лукічов В.В						
Затверд.		Лужецький						

Схема автентифікації за допомогою арифметичного методу



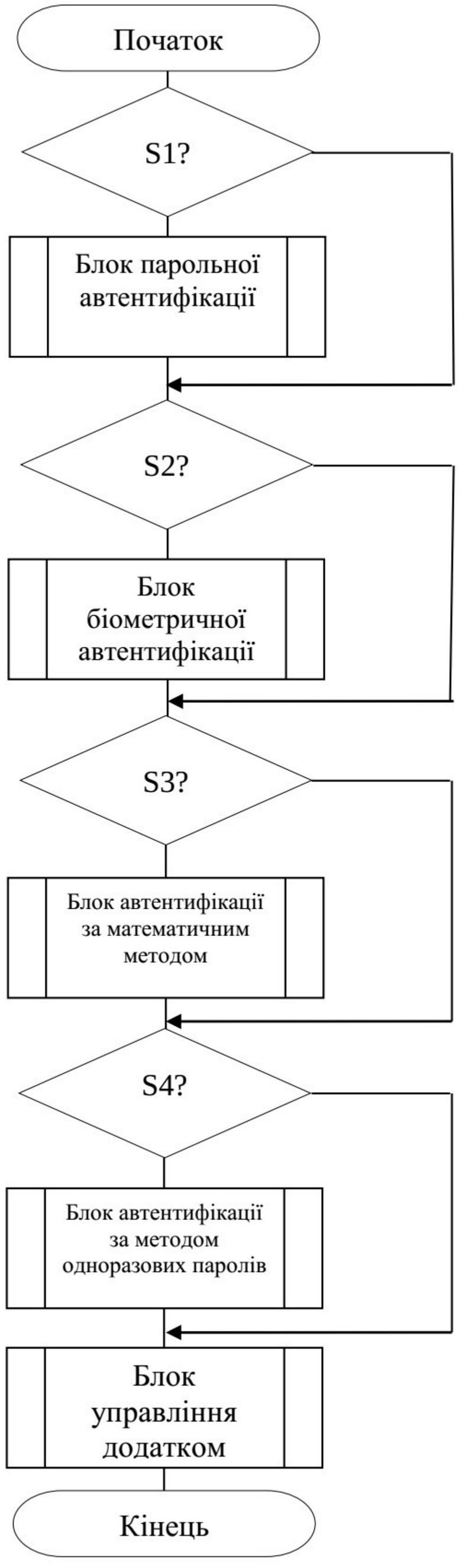
					08-20.МКР.002.00.000 142		
Зм	Арк.	№ докум.	Підпис	Дат			
Розроб.		Вітович М.М			Адаптивний метод автентифікації користувачів мобільних	Літ.	Маса / Масштаб
Перевір.		Лукічов В.В					
Реценз.		Крупельниць				ВНТУ, гр. 1 БС-	
Н. Контр.		Лукічов В.В					
Затверд.		Лужецький					

Схема автентифікації за допомогою метода одноразових паролів



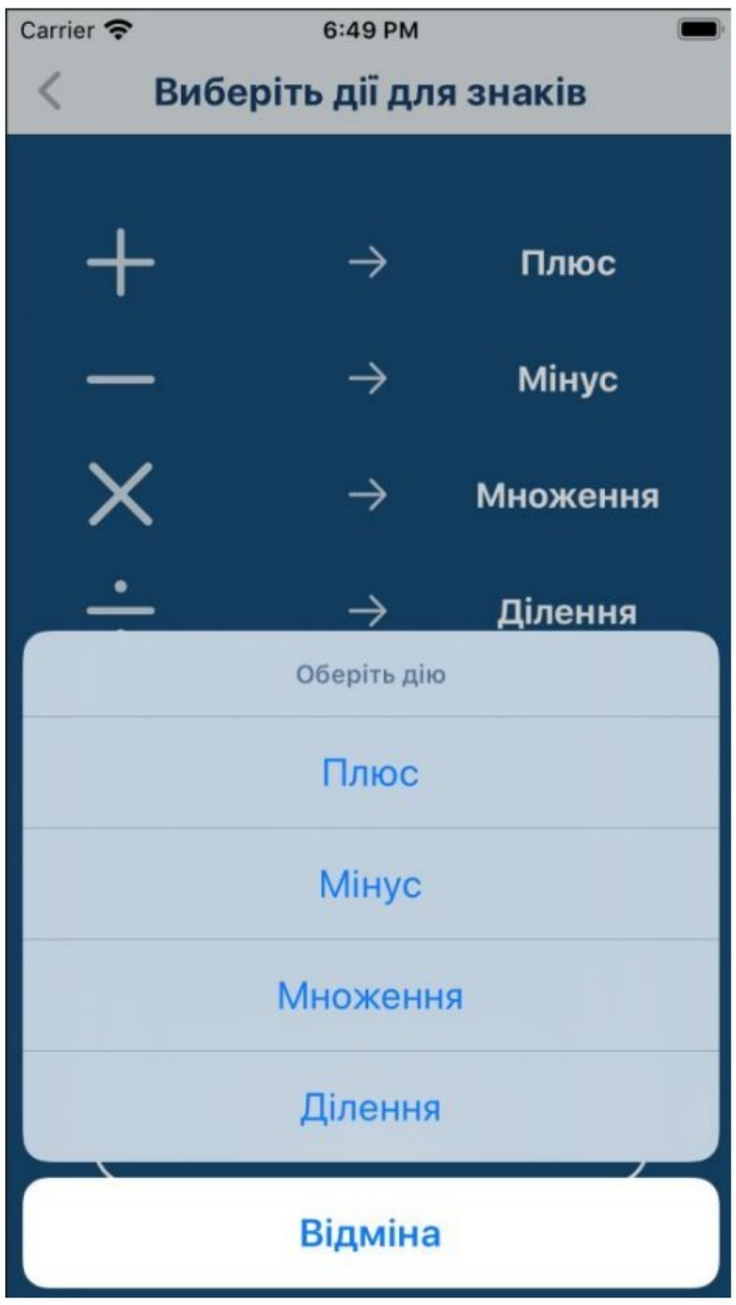
					08-20.МКР.002.00.000 143			
Зм	Арк.	№ докум.	Підпис	Дат				
Розроб.		Вітович М.М			Адаптивний метод автентифікації користувачів мобільних	Літ.	Маса	Масштаб
Перевір.		Лукічов В.В						
Реценз.		Крупельницць				ВНТУ. гр. 1 БС-		
Н. Контр.		Лукічов В.В						
Затверд.		Лужецький						

Схема автентифікації за допомогою комбінованого метода



					08-20.МКР.002.00.000 І44			
Зм	Арк.	№ докум.	Підпис	Дат				
Розроб.		Вітович М.М			Адаптивний метод автентифікації користувачів мобільних	Літ.	Маса	Масштаб
Перевір.		Лукічов В.В						
Реценз.		Крупельниць				ВНТУ, гр. 1 БС-		
Н. Контр.		Лукічов В.В						
Затверд.		Лужецький						

Фрагмент програмного засобу



					08-20.МКР.002.00.000 145	
Зм	Арк.	№ докум.	Підпис	Дат		
Розроб.		Вітович М.М			Адаптивний метод автентифікації користувачів мобільних	Літ. / Маса / Масштаб
Перевір.		Лукічов В.В				
Реценз.		Крупельницць				ВНТУ, гр. 1 БС-
Н. Контр.		Лукічов В.В				
Затверд.		Лужецький				